(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 March 2001 (29.03.2001)

PCT

(10) International Publication Number
WO 01/22650 A2

(51) International Patent Classification[7]: H04L 9/00

(21) International Application Number: PCT/US00/25796

(22) International Filing Date:
20 September 2000 (20.09.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/154,734 20 September 1999 (20.09.1999) US
60/200,396 27 April 2000 (27.04.2000) US

(71) Applicant: **ETHENTICA, INC.** [US/US]; 100 North Pointe Drive, Lake Forest, CA 92630 (US).

(72) Inventors: **DICKINSON, Alexander, G.**; 466 Monterey Drive, Laguna Beach, CA 92651 (US). **DAVENPORT, Roger, S.**; 14423 Heartside Place, Farmers Branch, TX 75234 (US). **OHARE, Mark, S.**; 22 Mountain Laurel, Dove Canyon, CA 92679 (US). **CLOUGH, Philip, W.**; 1306 South Greenstone Lane, Duncanville, TX 75137 (US). **ROHRBACH, Mark, D.**; 14685 Deerwood Street, Poway, CA 92064 (US). **CLAYTON, Richard, F.**; 7360 Kindler Road, Columbia, MD 21046 (US). **ZOCCOLI, James, G.**; 2114 Rheims 314, Carrollton, TX 75006 (US). **STARK, Gregory, H.**; Kirkwood Road, Millersville, MD 21108 (US). **ORSINI, Rick, L.**; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **FERRANTE, Michelle**; 5447 Vantage Pointe Road, Columbia, MD 21044 (US). **BROOKS, Aaron, A.**; 415 Brooks Court, Roanoke, TX 76262 (US).

(74) Agent: **ALTMAN, Daniel, E.**; Knobbe, Martens, Olson & Bear, LLP, 620 Newport Center Drive, 16th Floor, Newport Beach, CA 92660 (US).

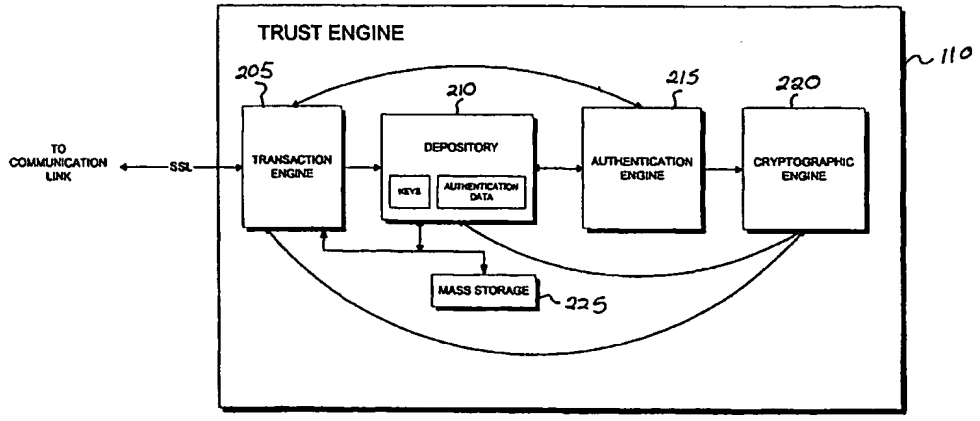
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, CZ (utility model), DE, DE (utility model), DK, DK (utility model), DM, DZ, EE, EE (utility model), ES, FI, FI (utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KR (utility model), KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (utility model), SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *Without international search report and to be republished upon receipt of that report.*

[Continued on next page]

(54) Title: SERVER-SIDE IMPLEMENTATION OF A CRYPTOGRAPHIC SYSTEM



(57) Abstract: The invention is a secure server, or trust engine, having server-centric keys, or in other words, storing cryptographic keys and user authentication data on a server. Users access cryptographic functionality through network access to the trust engine; however, the trust engine does not release actual cryptographic keys or other authentication data. Therefore, the system provides that the keys and data remain secure. The server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness, along with a wide variety of implementation possibilities.

WO 01/22650 A2

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SERVER-SIDE IMPLEMENTATION OF A CRYPTOGRAPHIC SYSTEM

Background of the Invention

Field of the Invention

The present invention relates in general to a server-side implementation of a cryptographic system and in particular to a trust engine for securely storing cryptographic keys and authentication data.

Description of the Related Art

In today's society, individuals and businesses conduct an ever-increasing amount of activities on and over computer systems. These computer systems, including proprietary and non-proprietary computer networks, are often storing, archiving, and transmitting all types of sensitive information. Thus, an ever-increasing need exists for ensuring data stored and transmitted over these systems cannot be read or otherwise compromised.

One common solution for securing computer systems is to provide login and password functionality. However, password management has proven to be quite costly with a large percentage of help desk calls relating to password issues. Moreover, passwords provide little security in that they are generally stored in a file susceptible to inappropriate access, through, for example, brute-force attacks.

Another solution for securing computer systems is to provide cryptographic infrastructures. Cryptography, in general, refers to protecting data by transforming, or encrypting, it into an unreadable format. Only those who possess the key(s) to the encryption can decrypt the data into a useable format. Cryptography is used to identify users, e.g., authentication, to allow access privileges, e.g., authorization, to create digital certificates and signatures, and the like. One popular cryptography system is a public-key system that uses two keys, a public key known to everyone and a private key known only to the individual or business owner thereof. Generally, the data encrypted with one key is decrypted with the other and neither key is recreatable from the other.

Unfortunately, even the foregoing typical public-key cryptographic systems are still highly reliant on the user for security. For example, cryptographic systems issue the private key to the user, for example, through the user's browser. Unsophisticated users then generally store the private key on a hard drive accessible to others through an open computer system, such as, for example, the Internet. On the other hand, users may choose poor names for files containing their private key, such as, for example, "key." The result of the foregoing and other acts is to allow the key or keys to be susceptible to compromise.

In addition to the foregoing compromises, a user may save his or her private key on a computer system configured with an archiving or backup system, potentially resulting in copies of the private key traveling through multiple computer storage devices or other systems. This security breach is often referred to as "key migration." Similar to key migration, many applications provide access to a user's private key through, at most, simple login and password access. As mentioned in the foregoing, login and password access often does not provide adequate security.

One solution for increasing the security of the foregoing cryptographic systems is to include biometrics as part of the authentication or authorization. Biometrics generally include measurable physical characteristics, such as, for example, finger prints or speech that can be checked by an automated system, such as, for example, pattern

matching or recognition of finger print patterns or speech patterns. In such systems, a user's biometric and/or keys may be stored on mobile computing devices, such as, for example, a smartcard, laptop, personal digital assistant, or mobile phone, thereby allowing the biometric or keys to be usable in a mobile environment.

The foregoing mobile biometric cryptographic system still suffers from a variety of drawbacks. For example, the mobile user may lose or break the smartcard or portable computing device, thereby having his or her access to potentially important data entirely cut-off. Alternatively, a malicious person may steal the mobile user's smartcard or portable computing device and use it to effectively steal the mobile user's digital credentials. On the other hand, the portable-computing device may be connected to an open system, such as the Internet, and, like passwords, the file where the biometric is stored may be susceptible to compromise through user inattentiveness to security or malicious intruders.

Summary of the Invention

Based on the foregoing, a need exists to provide a cryptographic system whose security is user-independent while still supporting mobile users. Accordingly, one aspect of the invention is to provide a secure server, or trust engine, having server-centric keys, or in other words, storing cryptographic keys and user authentication data on a server. According to this embodiment, a user accesses the trust engine in order to perform authentication and cryptographic functions, such as, for example, authentication, authorization, digital signing and generation, storage, and retrieval of certificates, encryption, notary-like and power-of-attorney-like actions, and the like.

Another aspect of the invention is to provide a reliable, or trusted, authentication process. Moreover, subsequent to a trustworthy positive authentication, a wide number of differing actions may be taken, from providing cryptographic technology, to system or device authorization and access, to permitting use or control of a wide number of electronic devices.

Another aspect of the invention is to provide cryptographic keys and authentication data in an environment where they are not lost, stolen, or compromised, thereby advantageously avoiding a need to continually reissue and manage new keys and authentication data. According to another aspect of the invention, the trust engine allows a user to use one key pair for multiple activities, vendors, and/or authentication requests. According to yet another aspect of the invention, the trust engine performs the majority of cryptographic processing, such as encrypting, authenticating, or signing, on the server side, thereby allowing clients to possess only minimal computing resources.

According to yet another aspect of the invention, the trust engine includes multiple depositories for storing portions of each cryptographic key and authentication data. The portions are created through a data splitting process that prohibits reconstruction without a predetermined portion from more than one depository. According to another embodiment, the multiple depositories are geographically remote such that a rogue employee or otherwise compromised system at one depository will not provide access to a user's key or authentication data.

According to yet another embodiment, the authentication process advantageously allows the trust engine to process vendor and user authentication activities in parallel. According to yet another embodiment, the trust engine

may advantageously track failed access attempts and thereby limit the number of times malicious intruders may attempt to subvert the system.

According to yet another embodiment, the trust engine may include multiple instantiations where each trust engine may predict and share processing loads with the others. According to yet another embodiment, the trust engine may include a redundancy module for polling a plurality of authentication results to ensure that more than one system authenticates the user.

Therefore, one aspect of the invention includes a remotely accessible secure cryptographic system for storing a plurality of private cryptographic keys to be associated with a plurality of users. The cryptographic system associates each of the plurality of users with one or more different keys from the plurality of private cryptographic keys and performs cryptographic functions for each user using the associated one or more different keys without releasing the plurality of private cryptographic keys to the users. The cryptographic system comprises a depository system having at least one server which stores a plurality of private cryptographic keys and a plurality of enrollment authentication data. Each enrollment authentication data identifies one of multiple users and each of the multiple users is associated with one or more different keys from the plurality of private cryptographic keys. The cryptographic system also comprises an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result. The cryptographic system also comprises a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using the associated one or more different keys received from the depository system. The cryptographic system also comprises a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

Another aspect of the invention includes a remotely accessible secure cryptographic system. The cryptographic system comprises a depository system having at least one server which stores at least one private key and a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of multiple users. The cryptographic system also comprises an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result. The cryptographic system also comprises a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using at least said private key received from the depository system. The cryptographic system also comprises a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

Another aspect of the invention includes a method of facilitating cryptographic functions. The method comprises associating a user from multiple users with one or more keys from a plurality of private cryptographic keys

stored on a secure server. The method also comprises receiving authentication data from the user, and comparing the authentication data to authentication data corresponding to the user, thereby verifying the identity of the user. The method also comprises utilizing the one or more keys to perform cryptographic functions without releasing the one or more keys to the user.

Another aspect of the invention includes an authentication system for uniquely identifying a user through secure storage of the user's enrollment authentication data. The authentication system comprises a plurality of data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores one of portions of enrollment authentication data. The authentication system also comprises an authentication engine which communicates with the plurality of data storage facilities. The authentication engine comprises a data splitting module which operates on the enrollment authentication data to create portions, a data assembling module which processes the portions from at least two of the data storage facilities to assemble the enrollment authentication data, and a data comparator module which receives current authentication data from a user and compares the current authentication data with the assembled enrollment authentication data to determine whether the user has been uniquely identified.

Another aspect of the invention includes a cryptographic system. The cryptographic system comprises a plurality of data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores one of portions of cryptographic keys. The cryptographic system also comprises a cryptographic engine which communicates with the plurality of data storage facilities. The cryptographic engine also comprises a data splitting module which operate on the cryptographic keys to create portions, a data assembling module which processes the portions from at least two of the data storage facilities to assemble the cryptographic keys, and a cryptographic handling module which receives the assembled cryptographic keys and performs cryptographic functions therewith.

Another aspect of the invention includes a method of storing authentication data in geographically remote secure data storage facilities thereby protecting the authentication data against comprise of any individual data storage facility. The method comprises receiving authentication data at a trust engine, combining at the trust engine the authentication data with a first substantially random value to form a first combined value, and combining the authentication data with a second substantially random value to form a second combined value. The method comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method comprises storing the second pairing in a second secure data storage facility remote from the first secure data storage facility.

Another aspect of the invention includes a method of storing authentication data comprising receiving authentication data, combining the authentication data with a first set of bits to form a second set of bits, and combining the authentication data with a third set of bits to form a fourth set of bits. The method also comprises creating a first pairing of the first set of bits with the third set of bits, The method also comprises creating a second

pairing of the first set of bits with the fourth set of bits, and storing one of the first and second pairings in a first computer accessible storage medium. The method also comprises storing the other of the first and second pairings in a second computer accessible storage medium.

Another aspect of the invention includes a method of storing cryptographic data in geographically remote secure data storage facilities thereby protecting the cryptographic data against comprise of any individual data storage facility. The method comprises receiving cryptographic data at a trust engine, combining at the trust engine the cryptographic data with a first substantially random value to form a first combined value, and combining the cryptographic data with a second substantially random value to form a second combined value. The method also comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method also comprises storing the second pairing in a secure second data storage facility remote from the first secure data storage facility.

Another aspect of the invention includes a method of storing cryptographic data comprising receiving authentication data and combining the cryptographic data with a first set of bits to form a second set of bits. The method also comprises combining the cryptographic data with a third set of bits to form a fourth set of bits, creating a first pairing of the first set of bits with the third set of bits, and creating a second pairing of the first set of bits with the fourth set of bits. The method also comprises storing one of the first and second pairings in a first computer accessible storage medium, and storing the other of the first and second pairings in a second computer accessible storage medium.

Another aspect of the invention includes a method of handling sensitive data in a cryptographic system, wherein the sensitive data exists in a useable form only during actions employing the sensitive data. The method also comprises receiving in a software module, substantially randomized sensitive data from a first computer accessible storage medium, and receiving in the software module, substantially randomized data from a second computer accessible storage medium. The method also comprises processing the substantially randomized sensitive data and the substantially randomized data in the software module to assemble the sensitive data and employing the sensitive data in a software engine to perform an action. The action includes one of authenticating a user and performing a cryptographic function.

Another aspect of the invention includes a secure authentication system. The secure authentication system comprises a plurality of authentication engines. Each authentication engine receives enrollment authentication data designed to uniquely identify a user to a degree of certainty. Each authentication engine receives current authentication data to compare to the enrollment authentication data, and each authentication engine determines an authentication result. The secure authentication system also comprises a redundancy system which receives the authentication result of at least two of the authentication engines and determines whether the user has been uniquely identified.

Brief Description of the Drawings

The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspects of an embodiment of the invention;

FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspects of an embodiment of the invention;

FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 7 illustrates a block diagram of a depository system, according to aspects of another embodiment of the invention;

FIGURE 8 illustrates a flow chart of a data splitting process according to aspects of an embodiment of the invention;

FIGURE 9A illustrates a data flow of an enrollment process according to aspects of an embodiment of the invention;

FIGURE 9B illustrates a flow chart of an interoperability process according to aspects of an embodiment of the invention;

FIGURE 10 illustrates a data flow of an authentication process according to aspects of an embodiment of the invention;

FIGURE 11 illustrates a data flow of a signing process according to aspects of an embodiment of the invention.

FIGURE 12 illustrates a data flow and an encryption/decryption process according to aspects and yet another embodiment of the invention;

FIGURE 13 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

FIGURE 14 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14, according to aspects of an embodiment of the invention;

FIGURE 16 illustrates a process for evaluating authentications according to one aspect of the invention;

FIGURE 17 illustrates a process for assigning a value to an authentication according to one aspect as shown in FIGURE 16 of the invention;

FIGURE 18 illustrates a process for performing trust arbitrage in an aspect of the invention as shown in FIGURE 17; and

FIGURE 19 illustrates a sample transaction between a user and a vendor according to aspects of an embodiment of the invention where an initial web based contact leads to a sales contract signed by both parties.

FIGURE 20 illustrates a sample user system with a cryptographic service provider module which provides security functions to a user system.

Detailed Description of the Preferred Embodiment

One aspect of the present invention is to provide a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. Users access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant. In addition, the cryptographic system may monitor agreements and determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device having attributes and features similar to those disclosed in U.S. Patent

Application No. 08/926,277, filed on September 5, 1997, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent Application No. 09/435,011, filed on November 5, 1999, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include a computer workstation, an interactive television, an interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile phone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, such as user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal data, such as mother's maiden name, favorite ice cream, or the like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to

periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as a fingerprint or speech, to the biometric device 107. The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting (often referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

A skilled artisan will recognize from the disclosure herein that the cryptographic keys may advantageously include some or all of symmetric keys, public keys, and private keys. In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like.

FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for example, PKCS7. According to one

embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate standard requested at the time of the request. The trust engine 110 will be disclosed in greater detail below.

FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system 120 advantageously comprises a Web server. Typical Web servers generally serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure is a global network of computers. The structure of the Internet, which is well known to those of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol (TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL).

Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110 also includes mass storage 225. As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing communications may include conventional SSL technologies.

According to one embodiment, the transaction engine 205 comprises a data routing device, such as a conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to

the one embodiment, the user system 105, the certificate authority 115, the vendor system 120, and the components of the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction engine 205 to properly route data throughout the cryptographic system.

Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110. Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vise-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

As the transaction engine 205 routes data to the various components of the cryptographic system 100, the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology. In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing advantageously having data requests and formats embedded therein.

As mentioned above, the depository 210 may advantageously comprises a plurality of secure data storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data

stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210 forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current authentication data and the enrollment authentication data for comparison.

According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption using public key technologies. For example, according to one embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220. According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305 having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may advantageously be secured through, for example, SSL technology.

FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and

cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with public key encryption coupled with conventional SSL technologies.

According to another embodiment, the depository 210 may comprise distinct and physically separated data storage facilities, as disclosed further with reference to FIGURE 7.

FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525. According to the preferred embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication engine 215 may advantageously compare conventional hashes of documents in order to render a comparison result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password which may only return a positive or negative match, a fingerprint may be determined to be a partial match, *e.g.* a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a transaction which has a low dollar value associated with it (*e.g.,* a $10 purchase) or a transaction with low risk (*e.g.,* admission to a members-only web site).

Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to proceed. Such transactions may include transactions of large dollar value (*e.g.,* signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (*e.g.,* remotely logging onto a government computer).

The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to impersonate an individual's authentication data to, for example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize and split the data into portions. According to one embodiment, original data is not recreatable from an individual portion. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 625 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or a wide number of alternative key management systems. In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number of functions generally surrounding cryptographic key management systems.

FIGURE 7 illustrates a simplified block diagram of a depository system 700 according to aspects of an embodiment of the invention. As shown in FIGURE 7, the depository system 700 advantageously comprises multiple data storage facilities, for example, data storage facilities D1, D2, D3, and D4. According to one embodiment of the invention, each of the data storage facilities D1 through D4 may advantageously comprise some or all of the elements disclosed with reference to the depository 210 of FIGURE 4. Similar to the depository 210, the data storage facilities D1 through D4 communicate with the transaction engine 205, the authentication engine 215, and the cryptographic engine 220, preferably through conventional SSL communication links transferring, for example, XML documents. Communications from the transaction engine 205 may advantageously include requests for data, wherein the request is advantageously broadcast to the IP address of each data storage facility D1 through D4. On the other hand, the transaction engine 205 may broadcast requests to particular data storage facilities based on a wide number of criteria, such as, for example, response time, server loads, maintenance schedules, or the like.

In response to requests for data from the transaction engine 205, the depository system 700 advantageously forwards stored data to the authentication engine 215 and the cryptographic engine 220. The respective data

assembling modules receive the forwarded data and assemble the data into useable formats. On the other hand, communications from the authentication engine 215 and the cryptographic engine 220 to the data storage facilities D1 through D4 may include the transmission of sensitive data to be stored. For example, according to one embodiment, the authentication engine 215 and the cryptographic engine 220 may advantageously employ their respective data splitting modules to divide sensitive data into undecipherable portions, and then transmit one or more undecipherable portions of the sensitive data to a particular data storage facility.

According to one embodiment, each data storage facility, D1 through D4, comprises a separate and independent storage system, such as, for example, a directory server. According to another embodiment of the invention, the depository system 700 comprises multiple geographically separated independent data storage systems. By distributing the sensitive data into distinct and independent storage facilities D1 through D4, some or all of which may be advantageously geographically separated, the depository system 700 provides redundancy along with additional security measures. For example, according to one embodiment, only data from two of the multiple data storage facilities, D1 through D4, are needed to decipher and reassemble the sensitive data. Thus, as many as two of the four data storage facilities D1 through D4 may be inoperative due to maintenance, system failure, power failure, or the like, without affecting the functionality of the trust engine 110. In addition, because, according to one embodiment, the data stored in each data storage facility is randomized and undecipherable, compromise of any individual data storage facility does not necessarily compromise the sensitive data. Moreover, in the embodiment having geographically separation of the data storage facilities, a compromise of multiple geographically remote facilities becomes increasingly difficult. In fact, even a rouge employee will be greatly challenged to subvert the needed multiple independent geographically remote data storage facilities.

Although the depository system 700 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the depository system 700. For example, the depository system 700 may comprise two or more data storage facilities. In addition, sensitive data may be mathematically operated such that portions from two or more data storage facilities are needed to reassemble and decipher the sensitive data.

As mentioned in the foregoing, the authentication engine 215 and the cryptographic engine 220 each include a data splitting module 520 and 610, respectively, for splitting sensitive data, such as, for example, the authentication data and the cryptographic key data. FIGURE 8 illustrates a flowchart of a data splitting process 800 performed by the data splitting module according to aspects of an embodiment of the invention. As shown in FIGURE 8, the data splitting process 800 begins at STEP 805 when sensitive data "S" is received by the data splitting module of the authentication engine 215 or the cryptographic engine 220. Preferably, in STEP 810, the data splitting module then generates a substantially random number, value, or string or set of bits, "A." For example, the random number A may be generated in a wide number of varying conventional techniques available to one of ordinary skill in the art, for producing high quality random numbers suitable for use in cryptographic applications. In addition, according to one embodiment, the random number A comprises a bit length equal to the bit length of the sensitive data, S.

In addition, in STEP 820 the data splitting process 800 generates another statistically random number "C." According to the preferred embodiment, the generation of the statistically random numbers A and C may advantageously be done in parallel. The data splitting module then combines the numbers A and C with the sensitive data S such that new numbers "B" and "D" are generated. For example, number B may comprise the binary combination of A XOR S and number D may comprise the binary combination of C XOR S. The foregoing combinations preferably occur in STEPS 825 and 830, respectively, and, according to one embodiment, the foregoing combinations also occur in parallel. The data splitting process 800 then proceeds to STEP 835 where the random numbers A and C and the numbers B and D are paired such that none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S. For example, the numbers may be paired as follows: AC, AD, BC, and BD. According to one embodiment, each of the foregoing pairings is distributed to one of the depositories D1 through D4 of FIGURE 7. According to another embodiment, each of the foregoing pairings is randomly distributed to one of the depositories D1 through D4. For example, during a first data splitting process 800, the pairing AC may be sent to depository D2, through, for example, a random selection of D2's IP address. Then, during a second data splitting process 800, the pairing AC may be sent to depository D4, through, for example, a random selection of D4's IP address.

Based on the foregoing, the data splitting process 800 advantageously places portions of the sensitive data in each of the four data storage facilities D1 through D4, such that no single data storage facility D1 through D4 includes sufficient encrypted data to recreate the original sensitive data S. As mentioned in the foregoing, such randomization of the data into individually unusable encrypted portions increases security and provides for maintained trust in the data even if one of the data storage facilities, D1 through D4, is compromised.

Although the data splitting process 800 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the data splitting process 800. For example, the data splitting process may advantageously split the data into two numbers, for example, random number A and number B and, randomly distribute A and B through two data storage facilities. Moreover, the data splitting process 800 may advantageously split the data among a wide number of data storage facilities through generation of an additional random numbers.

As mentioned in the foregoing, in order to recreate the sensitive data S, the data portions need to be derandomized and reorganized. This process may advantageously occur in the data assembling modules, 525 and 620, of the authentication engine 215 and the cryptographic engine 220, respectively. The data assembling module, for example, data assembly module 525, receives data portions from the data storage facilities D1 through D4, and reassembles the data into useable form. For example, according to one embodiment where the data splitting module 520 employed the data splitting process 800 of FIGURE 8, the data assembling module 525 uses data portions from at least two of the data storage facilities D1 through D4 to recreate the sensitive data S. For example, the pairings of AC, AD, BC, and BD, were distributed such that any two provide one of A and B, or, C and D. Noting that S = A XOR B or S = C XOR D indicates that when the data assembling module receives one of A and B, or, C and D,

the data assembling module 525 can advantageously reassemble the sensitive data S. Thus, the data assembling module 525 may assemble the sensitive data S, when, for example, it receives data portions from at least the first two of the data storage facilities D1 through D4 to respond to an assemble request by the trust engine 110.

Based on the above data splitting and assembling processes, the sensitive data S exists in usable format only in a limited area of the trust engine 110. For example, when the sensitive data S includes enrollment authentication data, usable, nonrandomized enrollment authentication data is available only in the authentication engine 215. Likewise, when the sensitive data S includes private cryptographic key data, usable, nonrandomized private cryptographic key data is available only in the cryptographic engine 220.

Although the data splitting and assembling processes are disclosed with reference to their preferred embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for splitting and reassembling the sensitive data S. For example, public-key encryption may be used to further secure the data at the data storage facilities D1 through D4.

FIGURE 9A illustrates a data flow of an enrollment process 900 according to aspects of an embodiment of the invention. As shown in FIGURE 9A, the enrollment process 900 begins at STEP 905 when a user desires to enroll with the trust engine 110 of the cryptographic system 100. According to this embodiment, the user system 105 advantageously includes a client-side applet, such as a Java-based, that queries the user to enter enrollment data, such as demographic data and enrollment authentication data. According to one embodiment, the enrollment authentication data includes user ID, password(s), biometric(s), or the like. According to one embodiment, during the querying process, the client-side applet preferably communicates with the trust engine 110 to ensure that a chosen user ID is unique. When the user ID is nonunique, the trust engine 110 may advantageously suggest a unique user ID. The client-side applet gathers the enrollment data and transmits the enrollment data, for example, through and XML document, to the trust engine 110, and in particular, to the transaction engine 205. According to one embodiment, the transmission is encoded with the public key of the authentication engine 215.

According to one embodiment, the user performs a single enrollment during STEP 905 of the enrollment process 900. For example, the user enrolls himself or herself as a particular person, such as Joe User. When Joe User desires to enroll as Joe User, CEO of Mega Corp., then according to this embodiment, Joe User enrolls a second time, receives a second unique user ID and the trust engine 110 does not associate the two identities. According to another embodiment of the invention, the enrollment process 900 provides for multiple user identities for a single user ID. Thus, in the above example, the trust engine 110 will advantageously associate the two identities of Joe User. As will be understood by a skilled artisan from the disclosure herein, a user may have many identities, for example, Joe User the head of household, Joe User the member of the Charitable Foundations, and the like. Even though the user may have multiple identities, according to this embodiment, the trust engine 110 preferably stores only one set of enrollment data. Moreover, users may advantageously add, edit/update, or delete identities as they are needed.

Although the enrollment process 900 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of

alternatives for gathering of enrollment data, and in particular, enrollment authentication data. For example, the applet may be common object model (COM) based applet or the like.

On the other hand, the enrollment process may include graded enrollment. For example, at a lowest level of enrollment, the user may enroll over the communication link 125 without producing documentation as to his or her identity. According to an increased level of enrollment, the user enrolls using a trusted third party, such as a digital notary. For example, and the user may appear in person to the trusted third party, produce credentials such as a birth certificate, driver's license, military ID, or the like, and the trusted third party may advantageously include, for example, their digital signature in enrollment submission. The trusted third party may include an actual notary, a government agency, such as the Post Office or Department of Motor Vehicles, a human resources person in a large company enrolling an employee, or the like. A skilled artisan will understand from the disclosure herein that a wide number of varying levels of enrollment may occur during the enrollment process 900.

After receiving the enrollment authentication data, at STEP 915, the transaction engine 205, using conventional FULL SSL technology forwards the enrollment authentication data to the authentication engine 215. In STEP 920, the authentication engine 215 decrypts the enrollment authentication data using the private key of the authentication engine 215. In addition, the authentication engine 215 employs the data splitting module to mathematically operate on the enrollment authentication data so as to split the data into at least two independently undecipherable, randomized, numbers. As mentioned in the foregoing, the at least two numbers may comprise a statistically random number and a binary XORed number. In STEP 925, the authentication engine 215 forwards each portion of the randomized numbers to one of the data storage facilities D1 through D4. As mentioned in the foregoing, the authentication engine 215 may also advantageously randomize which portions are transferred to which depositories.

Often during the enrollment process 900, the user will also desire to have a digital certificate issued such that he or she may receive encrypted documents from others outside the cryptographic system 100. As mentioned in the foregoing, the certificate authority 115 generally issues digital certificates according to one or more of several conventional standards. Generally, the digital certificate includes a public key of the user or system, which is known to everyone.

Whether the user requests a digital certificate at enrollment, or at another time, the request is transferred through the trust engine 110 to the authentication engine 215. According to one embodiment, the request includes an XML document having, for example, the proper name of the user. According to STEP 935, the authentication engine 215 transfers the request to the cryptographic engine 220 instructing the cryptographic engine 220 to generate a cryptographic key or key pair.

Upon request, at STEP 935, the cryptographic engine 220 generates at least one cryptographic key. According to one embodiment, the cryptographic handling module 625 generates a key pair, where one key is used as a private key, and one is used as a public key. The cryptographic engine 220 stores the private key and, according to one embodiment, a copy of the public key. In STEP 945, the cryptographic engine 220 transmits a request for a digital

certificate to the transaction engine 205. According to one embodiment, the request advantageously includes a standardized request, such as PKCS10, embedded in, for example, and XML document. The request for a digital certificate may advantageously correspond to one or more certificate authorities and the one or more standard formats the certificate authorities require.

In STEP 950 the transaction engine 205 forwards this request to the certificate authority 115, who, in STEP 955, returns a digital certificate. The return digital certificate may advantageously be in a standardized format, such as PKCS7, or in a proprietary format of one or more of the certificate authorities 115. In STEP 960, the digital certificate is received by the transaction engine 205, and a copy is forwarded to the user and a copy is stored with the trust engine 110. The trust engine 110 stores a copy of the certificate such that the trust engine 110 will not need to rely on the availability of the certificate authority 115. For example, when the user desires to send a digital certificate, or a third party requests the user's digital certificate, the request for the digital certificate is typically sent to the certificate authority 115. However, if the certificate authority 115 is conducting maintenance or has been victim of a failure or security compromise, the digital certificate may not be available.

At any time after issuing the cryptographic keys, the cryptographic engine 220 may advantageously employ the data splitting process 800 described above such that the cryptographic keys are split into independently undecipherable randomized numbers. Similar to the authentication data, at STEP 965 the cryptographic engine 220 transfers the randomized numbers to the data storage facilities D1 through D4.

A skilled artisan will recognize from the disclosure herein that the user may request a digital certificate anytime after enrollment. Moreover, the communications between systems may advantageously include FULL SSL or public-key encryption technologies. Moreover, the enrollment process may issue multiple digital certificates from multiple certificate authorities, including one or more proprietary certificate authorities internal or external to the trust engine 110.

As disclosed in STEPS 935 through 960, one embodiment of the invention includes the request for a certificate that is eventually stored on the trust engine 110. Because, according to one embodiment, the cryptographic handling module 625 issues the keys used by the trust engine 110, each certificate corresponds to a private key. Therefore, the trust engine 110 may advantageously provide for interoperability through monitoring the certificates owned by, or associated with, a user. For example, when the cryptographic engine 220 receives a request for a cryptographic function, the cryptographic handling module 625 may investigate the certificates owned by the requesting user to determine whether the user owns a private key matching the attributes of the request. When such a certificate exists, the cryptographic handling module 625 may use the certificate or the public or private keys associated therewith, to perform the requested function. When such a certificate does not exist, the cryptographic handling module 625 may advantageously and transparently perform a number of actions to attempt to remedy the lack of an appropriate key. For example, FIGURE 9B illustrates a flowchart of an interoperability process 970, which according to aspects of an embodiment of the invention, discloses the foregoing steps to ensure the cryptographic handling module 625 performs cryptographic functions using appropriate keys.

As shown in FIGURE 9B, the interoperability process 970 begins with STEP 972 where the cryptographic handling module 925 determines the type of certificate desired. According to one embodiment of the invention, the type of certificate may advantageously be specified in the request for cryptographic functions, or other data provided by the requestor. According to another embodiment, the certificate type may be ascertained by the data format of the request. For example, the cryptographic handling module 925 may advantageously recognize the request corresponds to a particular type.

According to one embodiment, the certificate type may include one or more algorithm standards, for example, RSA, ELGAMAL, or the like. In addition, the certificate type may include one or more key types, such as symmetric keys, public keys, strong encryption keys such as 256 bit keys, less secure keys, or the like. Moreover, the certificate type may include upgrades or replacements of one or more of the foregoing algorithm standards or keys, one or more message or data formats, one or more data encapsulation or encoding schemes, such as Base 32 or Base 64. The certificate type may also include compatibility with one or more third-party cryptographic applications or interfaces, one or more communication protocols, or one or more certificate standards or protocols. A skilled artisan will recognize from the disclosure herein that other differences may exist in certificate types, and translations to and from those differences may be implemented as disclosed herein.

Once the cryptographic handling module 625 determines the certificate type, the interoperability process 970 proceeds to STEP 974, and determines whether the user owns a certificate matching the type determined in STEP 974. When the user owns a matching certificate, for example, the trust engine 110 has access to the matching certificate through, for example, prior storage thereof, the cryptographic handling module 625 knows that a matching private key is also stored within the trust engine 110. For example, the matching private key may be stored within the depository 210 or depository system 700. The cryptographic handling module 625 may advantageously request the matching private key be assembled from, for example, the depository 210, and then in STEP 976, use the matching private key to perform cryptographic actions or functions. For example, as mentioned in the foregoing, the cryptographic handling module 625 may advantageously perform hashing, hash comparisons, data encryption or decryption, digital signature verification or creation, or the like.

When the user does not own a matching certificate, the interoperability process 970 proceeds to STEP 978 where the cryptographic handling module 625 determines whether the users owns a cross-certified certificate. According to one embodiment, cross-certification between certificate authorities occurs when a first certificate authority determines to trust certificates from a second certificate authority. In other words, the first certificate authority determines that certificates from the second certificate authority meets certain quality standards, and therefore, may be "certified" as equivalent to the first certificate authority's own certificates. Cross-certification becomes more complex when the certificate authorities issue, for example, certificates having levels of trust . For example, the first certificate authority may provide three levels of trust for a particular certificate, usually based on the degree of reliability in the enrollment process, while the second certificate authority may provide seven levels of trust. Cross-certification may advantageously track which levels and which the certificates from the second

certificate authority may be substituted for which levels and which certificates from the first. When the foregoing cross-certification is done officially and publicly between two certification authorities, the mapping of certificates and levels to one another is often called "chaining."

According to another embodiment of the invention, the cryptographic handling module 625 may advantageously develop cross-certifications outside those agreed upon by the certificate authorities. For example, the cryptographic handling module 625 may access a first certificate authority's certificate practice statement (CPS), or other published policy statement, and using, for example, the authentication tokens required by particular trust levels, match the first certificate authority's certificates to those of another certificate authority.

When, in STEP 978, the cryptographic handling module 625 determines that the users owns a cross-certified certificate, the interoperability process 970 proceeds to STEP 976, and performs the cryptographic action or function using the cross-certified public key, private key, or both. Alternatively, when the cryptographic handling module 625 determines that the users does not own a cross-certified certificate, the interoperability process 970 proceeds to STEP 980, where the cryptographic handling module 625 selects a certificate authority that issues the requested certificate type, or a certificate cross-certified thereto. In STEP 982, the cryptographic handling module 625 determines whether the user enrollment authentication data, discussed in the foregoing, meets the authentication requirements of the chosen certificate authority. For example, if the user enrolled over a network by, for example, answering demographic and other questions, the authentication data provided may establish a lower level of trust than a user providing biometric data and appearing before a third-party, such as, for example, a notary. According to one embodiment, the foregoing authentication requirements may advantageously be provided in the chosen authentication authority's CPS.

When the user has provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the interoperability process 970 proceeds to STEP 984, where the cryptographic handling module 625 acquires the certificate from the chosen certificate authority. According to one embodiment, the cryptographic handling module 625 acquires the certificate by following STEPS 945 through 960 of the enrollment process 900. For example, the cryptographic handling module 625 may advantageously employ one or more public keys from one or more of the key pairs already available to the cryptographic engine 220, to request the certificate from the certificate authority. According to another embodiment, the cryptographic handling module 625 may advantageously generate one or more new key pairs, and use the public keys corresponding thereto, to request the certificate from the certificate authority.

According to another embodiment, the trust engine 110 may advantageously include one or more certificate issuing modules capable of issuing one or more certificate types. According to this embodiment, the certificate issuing module may provide the foregoing certificate. When the cryptographic handling module 625 acquires the certificate, the interoperability process 970 proceeds to STEP 976, and performs the cryptographic action or function using the public key, private key, or both corresponding to the acquired certificate.

When the user, in STEP 982, has not provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the cryptographic handling module 625 determines, in STEP

986 whether there are other certificate authorities that have different authentication requirements. For example, the cryptographic handling module 625 may look for certificate authorities having lower authentication requirements, but still issue the chosen certificates, or cross-certifications thereof.

When the foregoing certificate authority having lower requirements exists, the interoperability process 970 proceeds to STEP 980 and chooses that certificate authority. Alternatively, when no such certificate authority exists, in STEP 988, the trust engine 110 may request additional authentication tokens from the user. For example, the trust engine 110 may request new enrollment authentication data comprising, for example, biometric data. Also, the trust engine 110 may request the user appear before a trusted third party and provide appropriate authenticating credentials, such as, for example, appearing before a notary with a drivers license, social security card, bank card, birth certificate, military ID, or the like. When the trust engine 110 receives updated authentication data, the interoperability process 970 proceeds to Step 984 and acquires the foregoing chosen certificate.

Through the foregoing interoperability process 970, the cryptographic handling module 625 advantageously provides seamless, transparent, translations and conversions between differing cryptographic systems. A skilled artisan will recognize from the disclosure herein, a wide number of advantages and implementations of the foregoing interoperable system. For example, the foregoing STEP 986 of the interoperability process 970 may advantageously include aspects of trust arbitrage, discussed in further detail below, where the certificate authority may under special circumstances accept lower levels of cross-certification. In addition, the interoperability process 970 may include ensuring interoperability between and employment of standard certificate revocations, such as employing certificate revocation lists (CRL), online certificate status protocols (OCSP), or the like.

FIGURE 10 illustrates a data flow of an authentication process 1000 according to aspects of an embodiment of the invention. According to one embodiment, the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user. For example, the authentication process 1000 begins at STEP 1005 where a user desires to perform a transaction with, for example, a vendor. Such transactions may include, for example, selecting a purchase option, requesting access to a restricted area or device of the vendor system 120, or the like. At STEP 1010, a vendor provides the user with a transaction ID and an authentication request. The transaction ID may advantageously include a 192 bit quantity having a 32 bit timestamp concatenated with a 128 bit random quantity, or a "nonce," concatenated with a 32 bit vendor specific constant. Such a transaction ID uniquely identifies the transaction such that copycat transactions can be refused by the trust engine 110.

The authentication request may advantageously include what level of authentication is needed for a particular transaction. For example, the vendor may specify a particular level of confidence that is required for the transaction at issue. If authentication cannot be made to this level of confidence, as will be discussed below, the transaction will not occur without either further authentication by the user to raise the level of confidence, or a change in the terms of the authentication between the vendor and the server. These issues are discussed more completely below.

According to one embodiment, the transaction ID and the authentication request may advantageously generated by a vendor-side applet or other software program. In addition, the transmission of the transaction ID and authentication data may include one or more XML documents encrypted using conventional SSL technology, such as, for example, ½ SSL, or, in other words vendor-side authenticated SSL.

After the user system 105 receives the transaction ID and authentication request, the user system 105 gathers the current authentication data, potentially including current biometric information, from the user. The user system 105, at STEP 1015, encrypts at least the current authentication data " B' " and the transaction ID, with the public key of the authentication engine 215, and transfers that data to the trust engine 110. The transmission preferably comprises XML documents encrypted with at least conventional ½ SSL technology. In STEP 1020, the transaction engine 205 receives the transmission, preferably recognizes the data format or request in the URL or URI, and forwards the transmission to the authentication engine 215.

During STEPS 1015 and 1020, the vendor system 120, at STEP 1025, forwards the transaction ID and the authentication request to the trust engine 110, using the preferred FULL SSL technology. This communication may also include a vendor ID, although vendor identification may also be communicated through a non-random portion of the transaction ID. At STEPS 1030 and 1035, the transaction engine 205 receives the communication, creates a record in the audit trail, and generates a request for the user's enrollment authentication data to be assembled from the data storage facilities D1 through D4. At STEP 1040, the depository system 700 transfers the portions of the enrollment authentication data corresponding to the user to the authentication engine 215. At STEP 1045, the authentication engine 215 decrypts the transmission using its private key and compares the enrollment authentication data to the current authentication data provided by the user.

The comparison of STEP 1045 may advantageously apply heuristical context sensitive authentication, as referred to in the forgoing, and discussed in further detail below. For example, if the biometric information received does not match perfectly, a lower confidence match results. In particular embodiments, the level of confidence of the authentication is balanced against the nature of the transaction and the desires of both the user and the vendor. Again, this is discussed in greater detail below.

At STEP 1050, the authentication engine 215 fills in the authentication request with the result of the comparison of STEP 1045. According to one embodiment of the invention, the authentication request is filled with a YES/NO or TRUE/FALSE result of the authentication process 1000. In STEP 1055 the filled-in authentication request is returned to the vendor for the vendor to act upon, for example, allowing the user to complete the transaction that initiated the authentication request. According to one embodiment, a confirmation message is passed to the user.

Based on the foregoing, the authentication process 1000 advantageously keeps sensitive data secure and produces results configured to maintain the integrity of the sensitive data. For example, the sensitive data is assembled only inside the authentication engine 215. For example, the enrollment authentication data is undecipherable until it is assembled in the authentication engine 215 by the data assembling module, and the current authentication data is undecipherable until it is unwrapped by the conventional SSL technology and the private key of

the authentication engine 215. Moreover, the authentication result transmitted to the vendor does not include the sensitive data, and the user may not even know whether he or she produced valid authentication data.

Although the authentication process 1000 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the authentication process 1000. For example, the vendor may advantageously be replaced by almost any requesting application, even those residing with the user system 105. For example, a client application, such as Microsoft Word, may use an application program interface (API) or a cryptographic API (CAPI) to request authentication before unlocking a document. Alternatively, a mail server, a network, a cellular phone, a personal or mobile computing device, a workstation, or the like, may all make authentication requests that can be filled by the authentication process 1000. In fact, after providing the foregoing trusted authentication process 1000, the requesting application or device may provide access to or use of a wide number of electronic or computer devices or systems.

Moreover, the authentication process 1000 may employ a wide number of alternative procedures in the event of authentication failure. For example, authentication failure may maintain the same transaction ID and request that the user reenter his or her current authentication data. As mentioned in the foregoing, use of the same transaction ID allows the comparator of the authentication engine 215 to monitor and limit the number of authentication attempts for a particular transaction, thereby creating a more secure cryptographic system 100.

In addition, the authentication process 1000 may be advantageously be employed to develop elegant single sign-on solutions, such as, unlocking a sensitive data vault. For example, successful or positive authentication may provide the authenticated user the ability to automatically access any number of passwords for an almost limitless number of systems and applications. For example, authentication of a user may provide the user access to password, login, financial credentials, or the like, associated with multiple online vendors, a local area network, various personal computing devices, Internet service providers, auction providers, investment brokerages, or the like. By employing a sensitive data vault, users may choose truly large and random passwords because they no longer need to remember them through association. Rather, the authentication process 1000 provides access thereto. For example, a user may choose a random alphanumeric string that is twenty plus digits in length rather than something associated with a memorable data, name, etc.

According to one embodiment, a sensitive data vault associated with a given user may advantageously be stored in the data storage facilities of the depository 210, or split and stored in the depository system 700. According to this embodiment, after positive user authentication, the trust engine 110 serves the requested sensitive data, such as, for example, to the appropriate password to the requesting application. According to another embodiment, the trust engine 110 may include a separate system for storing the sensitive data vault. For example, the trust engine 110 may include a stand-alone software engine implementing the data vault functionality and figuratively residing "behind" the foregoing front-end security system of the trust engine 110. According to this embodiment, the software engine

serves the requested sensitive data after the software engine receives a signal indicating positive user authentication from the trust engine 110.

In yet another embodiment, the data vault may be implemented by a third-party system. Similar to the software engine embodiment, the third-party system may advantageously serve the requested sensitive data after the third-party system receives a signal indicating positive user authentication from the trust engine 110. According to yet another embodiment, the data vault may be implemented on the user system 105. A user-side software engine may advantageously serve the foregoing data after receiving a signal indicating positive user authentication from the trust engine 110.

Although the foregoing data vaults are disclosed with reference to alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of additional implementations thereof. For example, a particular data vault may include aspects from some or all of the foregoing embodiments. In addition, any of the foregoing data vaults may employ one or more authentication requests at varying times. For example, any of the data vaults may require authentication every one or more transactions, periodically, every one ore more sessions, every access to one or more Webpages or Websites, at one or more other specified intervals, or the like.

FIGURE 11 illustrates a data flow of a signing process 1100 according to aspects of an embodiment of the invention. As shown in FIGURE 11, the signing process 1100 includes steps similar to those of the authentication process 1000 described in the foregoing with reference to FIGURE 10. According to one embodiment of the invention, the signing process 1100 first authenticates the user and then performs one or more of several digital signing functions as will be discussed in further detail below. According to another embodiment, the signing process 1100 may advantageously store data related thereto, such as hashes of messages or documents, or the like. This data may advantageously be used in an audit or any other event, such as for example, when a participating party attempts to repudiate a transaction.

As shown in FIGURE 11, during the authentication steps, the user and vendor may advantageously agree on a message, such as, for example, a contract. During signing, the signing process 1100 advantageously ensures that the contract signed by the user is identical to the contract supplied by the vendor. Therefore, according to one embodiment, during authentication, the vendor and the user include a hash of their respective copies of the message or contract, in the data transmitted to the authentication engine 215. By employing only a hash of a message or contract, the trust engine 110 may advantageously store a significantly reduced amount of data, providing for a more efficient and cost effective cryptographic system. In addition, the stored hash may be advantageously compared to a hash of a document in question to determine whether the document in question matches one signed by any of the parties. The ability to determine whether the document is identical to one relating to a transaction provides for additional evidence that can be used against a claim for repudiation by a party to a transaction.

In STEP 1103, the authentication engine 215 assembles the enrollment authentication data and compares it to the current authentication data provided by the user. When the comparator of the authentication engine 215 indicates that the enrollment authentication data matches the current authentication data, the comparator of the

authentication engine 215 also compares the hash of the message supplied by the vendor to the hash of the message supplied by the user. Thus, the authentication engine 215 advantageously ensures that the message agreed to by the user is identical to that agreed to by the vendor.

In STEP 1105, the authentication engine 215 transmits a digital signature request to the cryptographic engine 220. According to one embodiment of the invention, the request includes a hash of the message or contract. However, a skill artisan will recognize from the disclosure herein that the cryptographic engine 220 may encrypt virtually any text to form the desired digital signature. Returning to STEP 1105, the digital signature request preferably comprises an XML document communicated through conventional SSL technologies.

In STEP 1110, the authentication engine 215 transmits a request to each of the data storage facilities D1 through D4, such that each of the data storage facilities D1 through D4 transmit their respective portion of the cryptographic key or keys corresponding to a signing party. According to another embodiment, the cryptographic engine 220 employs some or all of the steps of the interoperability process 970 discussed in the foregoing, such that the cryptographic engine 220 first determines the appropriate key or keys to request from the depository 210 or the depository system 700 for the signing party, and takes actions to provide appropriate matching keys. According to still another embodiment, the authentication engine 215 or the cryptographic engine 220 may advantageously request one or more of the keys associated with the signing party and stored in the depository 210 or depository system 700.

According to one embodiment, the signing party includes one or both the user and the vendor. In such case, the authentication engine 215 advantageously requests the cryptographic keys corresponding to the user and/or the vendor. According to another embodiment, the signing party includes the trust engine 110. In this embodiment, the trust engine 110 is certifying that the authentication process 1000 properly authenticated the user, vendor, or both. Therefore, the authentication engine 215 requests the cryptographic key of the trust engine 110, such as, for example, the key belonging to the cryptographic engine 220, to perform the digital signature. According to another embodiment, the trust engine 110 performs a digital notary-like function. In this embodiment, the signing party includes the user, vendor, or both, along with the trust engine 110. Thus, the trust engine 110 provides the digital signature of the user and/or vendor, and then indicates with its own digital signature that the user and/or vendor were properly authenticated. In this embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the user, the vendor, or both. According to another embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the trust engine 110.

According to another embodiment, the trust engine 110 performs power of attorney-like functions. For example, the trust engine 110 may digitally sign the message on behalf of a third party. In such case, the authentication engine 215 requests the cryptographic keys associated with the third party. According to this embodiment, the signing process 1100 may advantageously include authentication of the third party, before allowing power of attorney-like functions. In addition, the authentication process 1000 may include a check for third party

constraints, such as, for example, business logic or the like dictating when and in what circumstances a particular third-party's signature may be used.

Based on the foregoing, in STEP 1110, the authentication engine requested the cryptographic keys from the data storage facilities D1 through D4 corresponding to the signing party. In STEP 1115, the data storage facilities D1 through D4 transmit their respective portions of the cryptographic key corresponding to the signing party to the cryptographic engine 220. According to one embodiment, the foregoing transmissions include SSL technologies. According to another embodiment, the foregoing transmissions may advantageously be super-encrypted with the public key of the cryptographic engine 220.

In STEP 1120, the cryptographic engine 220 assembles the foregoing cryptographic keys of the signing party and encrypts the message therewith, thereby forming the digital signature(s). In STEP 1125 of the signing process 1100, the cryptographic engine 220 transmits the digital signature(s) to the authentication engine 215. In STEP 1130, the authentication engine 215 transmits the filled-in authentication request along with a copy of the hashed message and the digital signature(s) to the transaction engine 205. In STEP 1135, the transaction engine 205 transmits a receipt comprising the transaction ID, an indication of whether the authentication was successful, and the digital signature(s), to the vendor. According to one embodiment, the foregoing transmission may advantageously include the digital signature of the trust engine 110. For example, the trust engine 110 may encrypt the hash of the receipt with its private key, thereby forming a digital signature to be attached to the transmission to the vendor.

According to one embodiment, the transaction engine 205 also transmits a confirmation message to the user.

Although the signing process 1100 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the signing process 1100. For example, the vendor may be replaced with a user application, such as an email application. For example, the user may wish to digitally sign a particular email with his or her digital signature. In such an embodiment, the transmission throughout the signing process 1100 may advantageously include only one copy of a hash of the message. Moreover, a skilled artisan will recognize from the disclosure herein that a wide number of client applications may request digital signatures. For example, the client applications may comprise word processors, spreadsheets, emails, voicemail, access to restricted system areas, or the like.

In addition, a skilled artisan will recognize from the disclosure herein that STEPS 1105 through 1120 of the signing process 1100 may advantageously employ some or all of the steps of the interoperability process 970 of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the digital signature under differing signature types.

FIGURE 12 illustrates a data flow of an encryption/decryption process 1200 according to aspects of an embodiment of the invention. As shown in FIGURE 12, the decryption process 1200 begins by authenticating the user using the authentication process 1000. According to one embodiment, the authentication process 1000 includes in the authentication request, a synchronous session key. For example, in conventional PKI technologies, it is understood by

skilled artisans that encrypting or decrypting data using public and private keys is mathematically intensive and may require significant system resources. However, in symmetric key cryptographic systems, or systems where the sender and receiver of a message share a single common key that is used to encrypt and decrypt a message, the mathematical operations are significantly simpler and faster. Thus, in the conventional PKI technologies, the sender of a message will generate synchronous session key, and encrypt the message using the simpler, faster symmetric key system. Then, the sender will encrypt the session key with the public key of the receiver. The encrypted session key will be attached to the synchronously encrypted message and both data are sent to the receiver. The receiver uses his or her private key to decrypt the session key, and then uses the session key to decrypt the message. Based on the foregoing, the simpler and faster symmetric key system is used for the majority of the encryption/decryption processing. Thus, in the decryption process 1200, the decryption advantageously assumes that a synchronous key has been encrypted with the public key of the user. Thus, as mentioned in the foregoing, the encrypted session key is included in the authentication request.

Returning to the decryption process 1200, after the user has been authenticated in STEP 1205, the authentication engine 215 forwards the encrypted session key to the cryptographic engine 220. In STEP 1210, the authentication engine 215 forwards a request to each of the data storage facilities, D1 through D4, requesting the cryptographic key data of the user. In STEP 1215, each data storage facility, D1 through D4, transmits their respective portion of the cryptographic key to the cryptographic engine 220. According to one embodiment, the foregoing transmission is encrypted with the public key of the cryptographic engine 220.

In STEP 1220 of the decryption process 1200, the cryptographic engine 220 assembles the cryptographic key and decrypts the session key therewith. In STEP 1225, the cryptographic engine forwards the session key to the authentication engine 215. In STEP 1227, the authentication engine 215 fills in the authentication request including the decrypted session key, and transmits the filled-in authentication request to the transaction engine 205. In STEP 1230, the transaction engine 205 forwards the authentication request along with the session key to the requesting application or vendor. Then, according to one embodiment, the requesting application or vendor uses the session key to decrypt the encrypted message.

Although the decryption process 1200 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the decryption process 1200. For example, the decryption process 1200 may forego synchronous key encryption and rely on full public-key technology. In such an embodiment, the requesting application may transmit the entire message to the cryptographic engine 220, or, may employ some type of compression or reversible hash in order to transmit the message to the cryptographic engine 220. A skilled artisan will also recognize from the disclosure herein that the foregoing communications may advantageously include XML documents wrapped in SSL technology.

The encryption/decryption process 1200 also provides for encryption of documents or other data. Thus, in STEP 1235, a requesting application or vendor may advantageously transmit to the transaction engine 205 of the trust engine 110, a request for the public key of the user. The requesting application or vendor makes this request

because the requesting application or vendor uses the public key of the user, for example, to encrypt the session key that will be used to encrypt the document or message. As mentioned in the enrollment process 900, the transaction engine 205 stores a copy of the digital certificate of the user, for example, in the mass storage 225. Thus, in STEP 1240 of the encryption process 1200, the transaction engine 205 requests the digital certificate of the user from the mass storage 225. In STEP 1245, the mass storage 225 transmits the digital certificate corresponding to the user, to the transaction engine 205. In STEP 1250, the transaction engine 205 transmits the digital certificate to the requesting application or vendor. According to one embodiment, the encryption portion of the encryption process 1200 does not include the authentication of a user. This is because the requesting vendor needs only the public key of the user, and is not requesting any sensitive data.

A skilled artisan will recognize from the disclosure herein that if a particular user does not have a digital certificate, the trust engine 110 may employ some or all of the enrollment process 900 in order to generate a digital certificate for that particular user. Then, the trust engine 110 may initiate the encryption/decryption process 1200 and thereby provide the appropriate digital certificate. In addition, a skilled artisan will recognize from the disclosure herein that STEPS 1220 and 1235 through 1250 of the encryption/decryption process 1200 may advantageously employ some or all of the steps of the interoperability process of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the encryption.

FIGURE 13 illustrates a simplified block diagram of a trust engine system 1300 according to aspects of yet another embodiment of the invention. As shown in FIGURE 13, the trust engine system 1300 comprises a plurality of distinct trust engines 1305, 1310, 1315, and 1320, respectively. To facilitate a more complete understanding of the invention, FIGURE 13 illustrates each trust engine, 1305, 1310, 1315, and 1320 as having a transaction engine, a depository, and an authentication engine. However, a skilled artisan will recognize that each transaction engine may advantageously comprise some or all of the elements and communication channels disclosed with reference to FIGURES 1-8. For example, one embodiment may advantageously include trust engines having transaction engines, depositories, and cryptographic servers.

According to one embodiment of the invention, each of the trust engines 1305, 1310, 1315 and 1320 are geographically separated, such that, for example, the trust engine 1305 may reside in a first location, the trust engine 1310 may reside in a second location, the trust engine 1315 may reside in a third location, and the trust engine 1320 may reside in a fourth location. The foregoing geographic separation advantageously decreases system response time while increasing the security of the overall trust engine system 1300.

For example, when a user logs onto the cryptographic system 100, the user may be nearest the first location and may desire to be authenticated. As described with reference to FIGURE 10, to be authenticated, the user provides current authentication data, such as a biometric or the like, and the current authentication data is compared to that user's enrollment authentication data. Therefore, according to one example, the user advantageously provides current authentication data to the geographically nearest trust engine 1305. The transaction engine 1321 of the trust engine 1305 then forwards the current authentication data to the authentication engine 1322 also residing at the first

location. According to another embodiment, the transaction engine 1321 forwards the current authentication data to one or more of the authentication engines of the trust engines 1310, 1315, or 1320.

The transaction engine 1321 also requests the assembly of the enrollment authentication data from the depositories of, for example, each of the trust engines, 1305 through 1320. According to this embodiment, each depository provides its portion of the enrollment authentication data to the authentication engine 1322 of the trust engine 1305. The authentication engine 1322 then employs the encrypted data portions from, for example, the first two depositories to respond, and assembles the enrollment authentication data into deciphered form. The authentication engine 1322 compares the enrollment authentication data with the current authentication data and returns an authentication result to the transaction engine 1321 of the trust engine 1305.

Based on the above, the trust engine system 1300 employs the nearest one of a plurality of geographically separated trust engines, 1305 through 1320, to perform the authentication process. According to one embodiment of the invention, the routing of information to the nearest transaction engine may advantageously be performed at client-side applets executing on one or more of the user system 105, vendor system 120, or certificate authority 115. According to an alternative embodiment, a more sophisticated decision process may be employed to select from the trust engines 1305 through 1320. For example, the decision may be based on the availability, operability, speed of connections, load, performance, geographic proximity, or a combination thereof, of a given trust engine.

In this way, the trust engine system 1300 lowers its response time while maintaining the security advantages associated with geographically remote data storage facilities, such as those discussed with reference to FIGURE 7 where each data storage facility stores randomized portions of sensitive data. For example, a security compromise at, for example, the depository 1325 of the trust engine 1315 does not necessarily compromise the sensitive data of the trust engine system 1300. This is because the depository 1325 contains only non-decipherable randomized data that, without more, is entirely useless.

According to another embodiment, the trust engine system 1300 may advantageously include multiple cryptographic engines arranged similar to the authentication engines. The cryptographic engines may advantageously perform cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment, the trust engine system 1300 may advantageously replace the multiple authentication engines with multiple cryptographic engines, thereby performing cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment of the invention, the trust engine system 1300 may replace each multiple authentication engine with an engine having some or all of the functionality of the authentication engines, cryptographic engines, or both, as disclosed in the foregoing.

Although the trust engine system 1300 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize that the trust engine system 1300 may comprise portions of trust engines 1305 through 1320. For example, the trust engine system 1300 may include one or more transaction engines, one or more depositories, one or more authentication engines, or one or more cryptographic engines.

FIGURE 14 illustrates a simplified block diagram of a trust engine system 1400 according to aspects of yet another embodiment of the invention. As shown in FIGURE 14, the trust engine system 1400 includes multiple trust engines 1405, 1410, 1415 and 1420. According to one embodiment, each of the trust engines 1405, 1410, 1415 and 1420, comprise some or all of the elements of trust engine 110 disclosed with reference to FIGURES 1-8. According to this embodiment, when the client side applets of the user system 105, the vendor system 120, or the certificate authority 115, communicate with the trust engine system 1400, those communications are sent to the IP address of each of the trust engines 1405 through 1420. Further, each transaction engine of each of the trust engines, 1405, 1410, 1415, and 1420, behaves similar to the transaction engine 1321 of the trust engine 1305 disclosed with reference to FIGURE 13. For example, during an authentication process, each transaction engine of each of the trust engines 1405, 1410, 1415, and 1420 transmits the current authentication data to their respective authentication engines and transmits a request to assemble the randomized data stored in each of the depositories of each of the trust engines 1405 through 1420. FIGURE 14 does not illustrate all of these communications, as such illustration would become overly complex. Continuing with the authentication process, each of the depositories then communicates its portion of the randomized data to each of the authentication engines of the each of the trust engines 1405 through 1420. Each of the authentication engines of the each of the trust engines employs its comparator to determine whether the current authentication data matches the enrollment authentication data provided by the depositories of each of the trust engines 1405 through 1420. According to this embodiment, the result of the comparison by each of the authentication engines is then transmitted to a redundancy module of the other three trust engines. For example, the result of the authentication engine from the trust engine 1405 is transmitted to the redundancy modules of the trust engines 1410, 1415, and 1420. Thus, the redundancy module of the trust engine 1405 likewise receives the result of the authentication engines from the trust engines 1410, 1415, and 1420.

FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14. The redundancy module comprises a comparator configured to receive the authentication result from three authentication engines and transmit that result to the transaction engine of the fourth trust engine. The comparator compares the authentication result form the three authentication engines, and if two of the results agree, the comparator concludes that the authentication result should match that of the two agreeing authentication engines. This result is then transmitted back to the transaction engine corresponding to the trust engine not associated with the three authentication engines.

Based on the foregoing, the redundancy module determines an authentication result from data received from authentication engines that are preferably geographically remote from the trust engine of that the redundancy module. By providing such redundancy functionality, the trust engine system 1400 ensures that a compromise of the authentication engine of one of the trust engines 1405 through 1420, is insufficient to compromise the authentication result of the redundancy module of that particular trust engine. A skilled artisan will recognize that redundancy module functionality of the trust engine system 1400 may also be applied to the cryptographic engine of each of the trust engines 1405 through 1420. However, such cryptographic engine communication was not shown in FIGURE 14

to avoid complexity. Moreover, a skilled artisan will recognize a wide number of alternative authentication result conflict resolution algorithms for the comparator of FIGURE 15.

According to yet another embodiment of the invention, the trust engine system 1400 may advantageously employ the redundancy module during cryptographic comparison steps. For example, some or all of the foregoing redundancy module disclosure with reference to FIGURES 14 and 15 may advantageously be implemented during a hash comparison of documents provided by one or more parties during a particular transaction.

Although the foregoing invention has been described in terms of certain preferred and alternative embodiments, other embodiments will be apparent to those of ordinary skill in the art from the disclosure herein. For example, the trust engine 110 may issue short-term certificates, where the private cryptographic key is released to the user for a predetermined period of time. For example, current certificate standards include a validity field that can be set to expire after a predetermined amount of time. Thus, the trust engine 110 may release a private key to a user where the private key would be valid for, for example, 24 hours. According to such an embodiment, the trust engine 110 may advantageously issue a new cryptographic key pair to be associated with a particular user and then release the private key of the new cryptographic key pair. Then, once the private cryptographic key is released, the trust engine 110 immediately expires any internal valid use of such private key, as it is no longer securable by the trust engine 110.

In addition, a skilled artisan will recognize that the cryptographic system 100 or the trust engine 110 may include the ability to recognize types of devices, such as a laptop, a cell phone, a network or the like. According to one embodiment, such recognition may come from data supplied in the request for a particular service, such as, a request for authentication leading to access or use, a request for cryptographic functionality, or the like. According to one embodiment, the foregoing request may include a unique device identifier, such as, for example, a processor ID. Alternatively, the request may include data in a particular recognizable data format. For example, mobile and satellite phones often do not include the processing power for full X509.v3 heavy encryption certificates, and therefore do not request them. According to this embodiment, the trust engine 110 may recognize the type of data format presented, and respond only in kind.

In an additional aspect of the system described above, context sensitive authentication can be provided using various techniques as will be described below. Context sensitive authentication, for example as shown in FIGURE 16, provides the possibility of evaluating not only the actual data which is sent by the user when attempting to authenticate himself, but also the circumstances surrounding the generation and delivery of that data. Such techniques may also support transaction specific trust arbitrage between the user and trust engine 110 or between the vendor and trust engine 110, as will be described below.

As discussed above, authentication is the process of proving that a user is who he says he is. Generally, authentication requires demonstrating some fact to an authentication authority. The trust engine 110 of the present invention represents the authority to which a user must authenticate himself. The user must demonstrate to the trust engine 110 that he is who he says he is by either: knowing something that only the user should know (knowledge-

based authentication), having something that only the user should have (token-based authentication), or by being something that only the user should be (biometric-based authentication).

Examples of knowledge-based authentication include without limitation a password, PIN number, or lock combination. Examples of token-based authentication include without limitation a house key, a physical credit card, a driver's license, or a particular phone number. Examples of biometric-based authentication include without limitation a fingerprint, a voice analysis, or a retinal scan.

Each type of authentication has particular advantages and disadvantages, and each provides a different level of security. For example, it is generally harder to create a false fingerprint that matches someone else's than it is to overhear someone's password and repeat it. Each type of authentication also requires a different type of data to be known to the authenticating authority in order to verify someone using that form of authentication.

As used herein, "authentication" will refer broadly to the overall process of verifying someone's identity to be who he says he is. An "authentication technique" will refer to a particular type of authentication based upon a particular piece of knowledge, physical token, or biometric reading. "Authentication data" refers to information which is sent to or otherwise demonstrated to an authentication authority in order to establish identity. "Enrollment data" will refer to the data which is initially submitted to an authentication authority in order to establish a baseline for comparison with authentication data. An "authentication instance" will refer to the data associated with an attempt to authenticate by an authentication technique.

The internal protocols and communications involved in the process of authenticating a user is described with reference to FIGURE 10 above. The part of this process within which the context sensitive authentication takes place occurs within the comparison step shown as STEP 1045 of FIGURE 10. This step takes place within the authentication engine 215 and involves assembling the enrollment data 410 retrieved from the depository 210 and comparing the authentication data provided by the user to it. One particular embodiment of this process is shown in FIGURE 16 and described below.

The current authentication data provided by the user and the enrollment data retrieved from the depository 210 are received by the authentication engine 215 in STEP 1600 of FIGURE 16. Both of these sets of data may contain data which is related to separate techniques of authentication. The authentication engine 215 separates the authentication data associated with each individual authentication instance in STEP 1605. This is necessary so that the authentication data is compared with the appropriate subset of the enrollment data for the user (*e.g.* fingerprint authentication data should be compared with fingerprint enrollment data, rather than password enrollment data).

Generally, authenticating a user involves one or more individual authentication instances, depending on which authentication techniques are available to the user. These methods are limited by the enrollment data which were provided by the user during his enrollment process (if the user did not provide a retinal scan when enrolling, he will not be able to authenticate himself using a retinal scan), as well as the means which may be currently available to the user (*e.g.* if the user does not have a fingerprint reader at his current location, fingerprint authentication will not be practical). In some cases, a single authentication instance may be sufficient to authenticate a user; however, in

certain circumstances a combination of multiple authentication instances may be used in order to more confidently authenticate a user for a particular transaction.

Each authentication instance consists of data related to a particular authentication technique (*e.g.* fingerprint, password, smart card, etc.) and the circumstances which surround the capture and delivery of the data for that particular technique. For example, a particular instance of attempting to authenticate via password will generate not only the data related to the password itself, but also circumstantial data, known as "metadata", related to that password attempt. This circumstantial data includes information such as: the time at which the particular authentication instance took place, the network address from which the authentication information was delivered, as well as any other information as is known to those of skill in the art which may be determined about the origin of the authentication data (the type of connection, the processor serial number, etc.).

In many cases, only a small amount of circumstantial metadata will be available. For example, if the user is located on a network which uses proxies or network address translation or another technique which masks the address of the originating computer, only the address of the proxy or router may be determined. Similarly, in many cases information such as the processor serial number will not be available because of either limitations of the hardware or operating system being used, disabling of such features by the operator of the system, or other limitations of the connection between the user's system and the trust engine 110.

As shown in FIGURE 16, once the individual authentication instances represented within the authentication data are extracted and separated in STEP 1605, the authentication engine 215 evaluates each instance for its reliability in indicating that the user is who he claims to be. The reliability for a single authentication instance will generally be determined based on several factors. These may be grouped as factors relating to the reliability associated with the authentication technique, which are evaluated in STEP 1610, and factors relating to the reliability of the particular authentication data provided, which are evaluated in STEP 1615. The first group includes without limitation the inherent reliability of the authentication technique being used, and the reliability of the enrollment data being used with that method. The second group includes without limitation the degree of match between the enrollment data and the data provided with the authentication instance, and the metadata associated with that authentication instance. Each of these factors may vary independently of the others.

The inherent reliability of an authentication technique is based on how hard it is for an imposter to provide someone else's correct data, as well as the overall error rates for the authentication technique. For passwords and knowledge based authentication methods, this reliability is often fairly low because there is nothing that prevents someone from revealing their password to another person and for that second person to use that password. Even a more complex knowledge based system may have only moderate reliability since knowledge may be transferred from person to person fairly easily. Token based authentication, such as having a proper smart card or using a particular terminal to perform the authentication, is similarly of low reliability used by itself, since there is no guarantee that the right person is in possession of the proper token.

However, biometric techniques are more inherently reliable because it is generally difficult to provide someone else with the ability to use your fingerprints in a convenient manner, even intentionally. Because subverting biometric authentication techniques is more difficult, the inherent reliability of biometric methods is generally higher than that of purely knowledge or token based authentication techniques. However, even biometric techniques may have some occasions in which a false acceptance or false rejection is generated. These occurrences may be reflected by differing reliabilities for different implementations of the same biometric technique. For example, a fingerprint matching system provided by one company may provide a higher reliability than one provided by a different company because one uses higher quality optics or a better scanning resolution or some other improvement which reduces the occurrence of false acceptances or false rejections.

Note that this reliability may be expressed in different manners. The reliability is desirably expressed in some metric which can be used by the heuristics 530 and algorithms of the authentication engine 215 to calculate the confidence level of each authentication. One preferred mode of expressing these reliabilities is as a percentage or fraction. For instance, fingerprints might be assigned an inherent reliability of 97%, while passwords might only be assigned an inherent reliability of 50%. Those of skill in the art will recognize that these particular values are merely exemplary and may vary between specific implementations.

The second factor for which reliability must be assessed is the reliability of the enrollment. This is part of the "graded enrollment" process referred to above. This reliability factor reflects the reliability of the identification provided during the initial enrollment process. For instance, if the individual initially enrolls in a manner where they physically produce evidence of their identity to a notary or other public official, and enrollment data is recorded at that time and notarized, the data will be more reliable than data which is provided over a network during enrollment and only vouched for by a digital signature or other information which is not truly tied to the individual.

Other enrollment techniques with varying levels of reliability include without limitation: enrollment at a physical office of the trust engine 110 operator; enrollment at a user's place of employment; enrollment at a post office or passport office; enrollment through an affiliated or trusted party to the trust engine 110 operator; anonymous or pseudonymous enrollment in which the enrolled identity is not yet identified with a particular real individual, as well as such other means as are known in the art.

These factors reflect the trust between the trust engine 110 and the source of identification provided during the enrollment process. For instance, if enrollment is performed in association with an employer during the initial process of providing evidence of identity, this information may be considered extremely reliable for purposes within the company, but may be trusted to a lesser degree by a government agency, or by a competitor. Therefore, trust engines operated by each of these other organizations may assign different levels of reliability to this enrollment.

Similarly, additional data which is submitted across a network, but which is authenticated by other trusted data provided during a previous enrollment with the same trust engine 110 may be considered as reliable as the original enrollment data was, even though the latter data were submitted across an open network. In such circumstances, a subsequent notarization will effectively increase the level of reliability associated with the original enrollment data. In

this way for example, an anonymous or pseudonymous enrollment may then be raised to a full enrollment by demonstrating to some enrollment official the identity of the individual matching the enrolled data.

The reliability factors discussed above are generally values which may be determined in advance of any particular authentication instance. This is because they are based upon the enrollment and the technique, rather than the actual authentication. In one embodiment, the step of generating reliability based upon these factors involves looking up previously determined values for this particular authentication technique and the enrollment data of the user. In a further aspect of an advantageous embodiment of the present invention, such reliabilities may be included with the enrollment data itself. In this way, these factors are automatically delivered to the authentication engine 215 along with the enrollment data sent from the depository 210.

While these factors may generally be determined in advance of any individual authentication instance, they still have an effect on each authentication instance which uses that particular technique of authentication for that user. Furthermore, although the values may change over time (*e.g.* if the user re-enrolls in a more reliable fashion), they are not dependent on the authentication data itself. By contrast, the reliability factors associated with a single specific instance's data may vary on each occasion. These factors, as discussed below, must be evaluated for each new authentication in order to generate reliability scores in STEP 1615.

The reliability of the authentication data reflects the match between the data provided by the user in a particular authentication instance and the data provided during the authentication enrollment. This is the fundamental question of whether the authentication data matches the enrollment data for the individual the user is claiming to be. Normally, when the data do not match, the user is considered to not be successfully authenticated, and the authentication fails. The manner in which this is evaluated may change depending on the authentication technique used. The comparison of such data is performed by the comparator 515 function of the authentication engine 215 as shown in FIGURE 5.

For instance, matches of passwords are generally evaluated in a binary fashion. In other words, a password is either a perfect match, or a failed match. It is usually not desirable to accept as even a partial match a password which is close to the correct password if it is not exactly correct. Therefore, when evaluating a password authentication, the reliability of the authentication returned by the comparator 515 is typically either 100% (correct) or 0% (wrong), with no possibility of intermediate values.

Similar rules to those for passwords are generally applied to token based authentication methods, such as smart cards. This is because having a smart card which has a similar identifier or which is similar to the correct one, is still just as wrong as having any other incorrect token. Therefore tokens tend also to be binary authenticators: a user either has the right token, or he doesn't.

However, certain types of authentication data, such as questionnaires and biometrics, are generally not binary authenticators. For example, a fingerprint may match a reference fingerprint to varying degrees. To some extent, this may be due to variations in the quality of the data captured either during the initial enrollment or in subsequent authentications. (A fingerprint may be smudged or a person may have a still healing scar or burn on a

particular finger.) In other instances the data may match less than perfectly because the information itself is somewhat variable and based upon pattern matching. (A voice analysis may seem close but not quite right because of background noise, or the acoustics of the environment in which the voice is recorded, or because the person has a cold.) Finally, in situations where large amounts of data are being compared, it may simply be the case that much of the data matches well, but some doesn't. (A ten-question questionnaire may have resulted in eight correct answers to personal questions, but two incorrect answers.) For any of these reasons, the match between the enrollment data and the data for a particular authentication instance may be desirably assigned a partial match value by the comparator 515. In this way, the fingerprint might be said to be a 85% match, the voice print a 65% match, and the questionnaire an 80% match, for example.

This measure (degree of match) produced by the comparator 515 is the factor representing the basic issue of whether an authentication is correct or not. However, as discussed above, this is only one of the factors which may be used in determining the reliability of a given authentication instance. Note also that even though a match to some partial degree may be determined, that ultimately, it may be desirable to provide a binary result based upon a partial match. In an alternate mode of operation, it is also possible to treat partial matches as binary, *i.e.* either perfect (100%) or failed (0%) matches, based upon whether or not the degree of match passes a particular threshold level of match. Such a process may be used to provide a simple pass/fail level of matching for systems which would otherwise produce partial matches.

Another factor to be considered in evaluating the reliability of a given authentication instance concerns the circumstances under which the authentication data for this particular instance are provided. As discussed above, the circumstances refer to the metadata associated with a particular authentication instance. This may include without limitation such information as: the network address of the authenticator, to the extent that it can be determined; the time of the authentication; the mode of transmission of the authentication data (phone line, cellular, network, etc.); and the serial number of the system of the authenticator.

These factors can be used to produce a profile of the type of authentication that is normally requested by the user. Then, this information can be used to assess reliability in at least two manners. One manner is to consider whether the user is requesting authentication in a manner which is consistent with the normal profile of authentication by this user. If the user normally makes authentication requests from one network address during business days (when she is at work) and from a different network address during evenings or weekends (when she is at home), an authentication which occurs from the home address during the business day is less reliable because it is outside the normal authentication profile. Similarly, if the user normally authenticates using a fingerprint biometric and in the evenings, an authentication which originates during the day using only a password is less reliable.

An additional way in which the circumstantial metadata can be used to evaluate the reliability of an instance of authentication is to determine how much corroboration the circumstance provides that the authenticator is the individual he claims to be. For instance, if the authentication comes from a system with a serial number known to be associated with the user, this is a good circumstantial indicator that the user is who they claim to be. Conversely, if

the authentication is coming from a network address which is known to be in Los Angeles when the user is known to reside in London, this is an indication that this authentication is less reliable based on its circumstances.

It is also possible that a cookie or other electronic data may be placed upon the system being used by a user when they interact with a vendor system or with the trust engine 110. This data is written to the storage of the system of the user and may contain an identification which may be read by a Web browser or other software on the user system. If this data is allowed to reside on the user system between sessions (a "persistent cookie"), it may be sent with the authentication data as further evidence of the past use of this system during authentication of a particular user. In effect, the metadata of a given instance, particularly a persistent cookie, may form a sort of token based authenticator itself.

Once the appropriate reliability factors based on the technique and data of the authentication instance are generated as described above in STEPS 1610 and 1615 respectively, they are used to produce an overall reliability for the authentication instance provided in STEP 1620. One means of doing this is simply to express each reliability as a percentage and then to multiply them together.

For example, suppose the authentication data is being sent in from a network address known to be the user's home computer completely in accordance with the user's past authentication profile (100%), and the technique being used is fingerprint identification (97%), and the initial finger print data was registered through the user's employer with the trust engine 110 (90%), and the match between the authentication data and the original fingerprint template in the enrollment data is very good (99%). The overall reliability of this authentication instance could then be calculated as the product of these reliabilities: 100% * 97% * 90% * 99% = 86.4% reliability.

This calculated reliability represents the reliability of one single instance of authentication. The overall reliability of a single authentication instance may also be calculated using techniques which treat the different reliability factors differently, for example by using formulas where different weights are assigned to each reliability factor. Furthermore, those of skill in the art will recognize that the actual values used may represent values other than percentages and may use non-arithmetic systems. One embodiment may include a module used by an authentication requestor to set the weights for each factor and the algorithms used in establishing the overall reliability of the authentication instance.

The authentication engine 215 may use the above techniques and variations thereof to determine the reliability of a single authentication instance, indicated as STEP 1620. However, it may be useful in many authentication situations for multiple authentication instances to be provided at the same time. For example, while attempting to authenticate himself using the system of the present invention, a user may provide a user identification, fingerprint authentication data, a smart card, and a password. In such a case, three independent authentication instances are being provided to the trust engine 110 for evaluation. Proceeding to STEP 1625, if the authentication engine 215 determines that the data provided by the user includes more than one authentication instance, then each instance in turn will be selected as shown in STEP 1630 and evaluated as described above in STEPS 1610, 1615 and 1620.

Note that many of the reliability factors discussed may vary from one of these instances to another. For instance, the inherent reliability of these techniques is likely to be different, as well as the degree of match provided between the authentication data and the enrollment data. Furthermore, the user may have provided enrollment data at different times and under different circumstances for each of these techniques, providing different enrollment reliabilities for each of these instances as well. Finally, even though the circumstances under which the data for each of these instances is being submitted is the same, the use of such techniques may each fit the profile of the user differently, and so may be assigned different circumstantial reliabilities. (For example, the user may normally use their password and fingerprint, but not their smart card.)

As a result, the final reliability for each of these authentication instances may be different from one another. However, by using multiple instances together, the overall confidence level for the authentication will tend to increase.

Once the authentication engine has performed STEPS 1610 through 1620 for all of the authentication instances provided in the authentication data, the reliability of each instance is used in STEP 1635 to evaluate the overall authentication confidence level. This process of combining the individual authentication instance reliabilities into the authentication confidence level may be modeled by various methods relating the individual reliabilities produced, and may also address the particular interaction between some of these authentication techniques. (For example, multiple knowledge-based systems such as passwords may produce less confidence than a single password and even a fairly weak biometric, such as a basic voice analysis.)

One means in which the authentication engine 215 may combine the reliabilities of multiple concurrent authentication instances to generate a final confidence level is to multiply the unreliability of each instance to arrive at a total unreliability. The unreliability is generally the complementary percentage of the reliability. For example, a technique which is 84% reliable is 16% unreliable. The three authentication instances described above (fingerprint, smart card, password) which produce reliabilities of 86%, 75%, and 72% would have corresponding unreliabilities of (100 - 86)%, (100 - 75)% and (100 - 72)%, or 14%, 25%, and 28%, respectively. By multiplying these unreliabilities, we get a cumulative unreliability of 14% * 25% * 28% = .98% unreliability, which corresponds to a reliability of 99.02%.

In an additional mode of operation, additional factors and heuristics 530 may be applied within the authentication engine 215 to account for the interdependence of various authentication techniques. For example, if someone has unauthorized access to a particular home computer, they probably have access to the phone line at that address as well. Therefore, authenticating based on an originating phone number as well as upon the serial number of the authenticating system does not add much to the overall confidence in the authentication. However, knowledge based authentication is largely independent of token based authentication (*i.e.* if someone steals your cellular phone or keys, they are no more likely to know your PIN or password than if they hadn't).

Furthermore, different vendors or other authentication requestors may wish to weigh different aspects of the authentication differently. This may include the use of separate weighing factors or algorithms used in calculating the

reliability of individual instances as well as the use of different means to evaluate authentication events with multiple instances.

For instance, vendors for certain types of transactions, for instance corporate email systems, may desire to authenticate primarily based upon heuristics and other circumstantial data by default. Therefore, they may apply high weights to factors related to the metadata and other profile related information associated with the circumstances surrounding authentication events. This arrangement could be used to ease the burden on users during normal operating hours, by not requiring more from the user than that he be logged on to the correct machine during business hours. However, another vendor may weigh authentications coming from a particular technique most heavily, for instance fingerprint matching, because of a policy decision that such a technique is most suited to authentication for the particular vendor's purposes.

Such varying weights may be defined by the authentication requestor in generating the authentication request and sent to the trust engine 110 with the authentication request in one mode of operation. Such options could also be set as preferences during an initial enrollment process for the authentication requestor and stored within the authentication engine in another mode of operation.

Once the authentication engine 215 produces an authentication confidence level for the authentication data provided, this confidence level is used to complete the authentication request in STEP 1640, and this information is forwarded from the authentication engine 215 to the transaction engine 205 for inclusion in a message to the authentication requestor.

The process described above is merely exemplary, and those of skill in the art will recognize that the steps need not be performed in the order shown. Furthermore, certain steps, such as the evaluation of the reliability of each authentication instance provided, may be carried out in parallel with one another if circumstances permit.

In a further aspect of this invention, a method is provided to accommodate conditions when the authentication confidence level produced by the process described above fails to meet the required trust level of the vendor or other party requiring the authentication. In circumstances such as these where a gap exists between the level of confidence provided and the level of trust desired, the operator of the trust engine 110 is in a position to provide opportunities for one or both parties to provide alternate data or requirements in order to close this trust gap. This process will be referred to as "trust arbitrage" herein.

Trust arbitrage may take place within a framework of cryptographic authentication as described above with reference to FIGURES 10 and 11. As shown therein, a vendor or other party will request authentication of a particular user in association with a particular transaction. In one circumstance, the vendor simply requests an authentication, either positive or negative, and after receiving appropriate data from the user, the trust engine 110 will provide such a binary authentication. In circumstances such as these, the degree of confidence required in order to secure a positive authentication is determined based upon preferences set within the trust engine 110.

However, it is also possible that the vendor may request a particular level of trust in order to complete a particular transaction. This required level may be included with the authentication request (*e.g.* authenticate this user

to 96% confidence) or may be determined by the trust engine 110 based on other factors associated with the transaction (*i.e.* authenticate this user as appropriate for this transaction). One such factor might be the economic value of the transaction. For transactions which have greater economic value, a higher degree of trust may be required. Similarly, for transactions with high degrees of risk a high degree of trust may be required. Conversely, for transactions which are either of low risk or of low value, lower trust levels may be required by the vendor or other authentication requestor.

The process of trust arbitrage occurs between the steps of the trust engine 110 receiving the authentication data in STEP 1050 of FIGURE 10 and the return of an authentication result to the vendor in STEP 1055 of FIGURE 10. Between these steps, the process which leads to the evaluation of trust levels and the potential trust arbitrage occurs as shown in FIGURE 17. In circumstances where simple binary authentication is performed, the process shown in FIGURE 17 reduces to having the transaction engine 205 directly compare the authentication data provided with the enrollment data for the identified user as discussed above with reference to FIGURE 10, flagging any difference as a negative authentication.

As shown in FIGURE 17, the first step after receiving the data in STEP 1050 is for the transaction engine 205 to determine the trust level which is required for a positive authentication for this particular transaction in STEP 1710. This step may be performed by one of several different methods. The required trust level may be specified to the trust engine 110 by the authentication requestor at the time when the authentication request is made. The authentication requestor may also set a preference in advance which is stored within the depository 210 or other storage which is accessible by the transaction engine 205. This preference may then be read and used each time an authentication request is made by this authentication requestor. The preference may also be associated with a particular user as a security measure such that a particular level of trust is always required in order to authenticate that user, the user preference being stored in the depository 210 or other storage media accessible by the transaction engine 205. The required level may also be derived by the transaction engine 205 or authentication engine 215 based upon information provided in the authentication request, such as the value and risk level of the transaction to be authenticated.

In one mode of operation, a policy management module or other software which is used when generating the authentication request is used to specify the required degree of trust for the authentication of the transaction. This may be used to provide a series of rules to follow when assigning the required level of trust based upon the policies which are specified within the policy management module. One advantageous mode of operation is for such a module to be incorporated with the web server of a vendor in order to appropriately determine required level of trust for transactions initiated with the vendor's web server. In this way, transaction requests from users may be assigned a required trust level in accordance with the policies of the vendor and such information may be forwarded to the trust engine 110 along with the authentication request.

This required trust level correlates with the degree of certainty that the vendor wants to have that the individual authenticating is in fact who he identifies himself as. For example, if the transaction is one where the

vendor wants a fair degree of certainty because goods are changing hands, the vendor may require a trust level of 85%. For situation where the vendor is merely authenticating the user to allow him to view members only content or exercise privileges on a chat room, the downside risk may be small enough that the vendor requires only a 60% trust level. However, to enter into a production contract with a value of tens of thousands of dollars, the vendor may require a trust level of 99% or more.

This required trust level represents a metric to which the user must authenticate himself in order to complete the transaction. If the required trust level is 85% for example, the user must provide authentication to the trust engine 110 sufficient for the trust engine 110 to say with 85% confidence that the user is who they say they are. It is the balance between this required trust level and the authentication confidence level which produces either a positive authentication (to the satisfaction of the vendor) or a possibility of trust arbitrage.

As shown in FIGURE 17, after the transaction engine 205 receives the required trust level, it compares in STEP 1720 the required trust level to the authentication confidence level which the authentication engine 215 calculated for the current authentication (as discussed with reference to FIGURE 16). If the authentication confidence level is higher than the required trust level for the transaction in STEP 1730, then the process moves to STEP 1740 where a positive authentication for this transaction is produced by the transaction engine 205. A message to this effect will then be inserted into the authentication results and returned to the vendor by the transaction engine 205 as shown in STEP 1055 (see FIGURE 10).

However, if the authentication confidence level does not fulfill the required trust level in STEP 1730, then a confidence gap exists for the current authentication, and trust arbitrage is conducted in STEP 1750. Trust arbitrage is described more completely with reference to FIGURE 18 below. This process as described below takes place within the transaction engine 205 of the trust engine 110. Because no authentication or other cryptographic operations are needed to execute trust arbitrage (other than those required for the SSL communication between the transaction engine 205 and other components), the process may be performed outside the authentication engine 215. However, as will be discussed below, any reevaluation of authentication data or other cryptographic or authentication events will require the transaction engine 205 to resubmit the appropriate data to the authentication engine 215. Those of skill in the art will recognize that the trust arbitrage process could alternately be structured to take place partially or entirely within the authentication engine 215 itself.

As mentioned above, trust arbitrage is a process where the trust engine 110 mediates a negotiation between the vendor and user in an attempt to secure a positive authentication where appropriate. As shown in STEP 1805, the transaction engine 205 first determines whether or not the current situation is appropriate for trust arbitrage. This may be determined based upon the circumstances of the authentication, *e.g.* whether this authentication has already been through multiple cycles of arbitrage, as well as upon the preferences of either the vendor or user, as will be discussed further below.

In such circumstances where arbitrage is not possible, the process proceeds to STEP 1810 where the transaction engine 205 generates a negative authentication and then inserts it into the authentication results which

are sent to the vendor in STEP 1055 (see FIGURE 10). One limit which may be advantageously used to prevent authentications from pending indefinitely is to set a time-out period from the initial authentication request. In this way, any transaction which is not positively authenticated within the time limit is denied further arbitrage and negatively authenticated. Those of skill in the art will recognize that such a time limit may vary depending upon the circumstances of the transaction and the desires of the user and vendor. Limitations may also be placed upon the number of attempts that may be made at providing a successful authentication. Such limitations may be handled by an attempt limiter 535 as shown in FIGURE 5.

If arbitrage is not prohibited in STEP 1805, the transaction engine 205 will then engage in negotiation with one or both of the transacting parties. The transaction engine 205 may send a message to the user requesting some form of additional authentication in order to boost the authentication confidence level produced as shown in STEP 1820. In the simplest form, this may simply indicates that authentication was insufficient. A request to produce one or more additional authentication instances to improve the overall confidence level of the authentication may also be sent.

If the user provides some additional authentication instances in STEP 1825, then the transaction engine 205 adds these authentication instances to the authentication data for the transaction and forwards it to the authentication engine 215 as shown in STEP 1015 (see FIGURE 10), and the authentication is reevaluated based upon both the pre-existing authentication instances for this transaction and the newly provided authentication instances.

An additional type of authentication may be a request from the trust engine 110 to make some form of person-to-person contact between the trust engine 110 operator (or a trusted associate) and the user, for example, by phone call. This phone call or other non-computer authentication can be used to provide personal contact with the individual and also to conduct some form of questionnaire based authentication. This also may give the opportunity to verify an originating telephone number and potentially a voice analysis of the user when he calls in. Even if no additional authentication data can be provided, the additional context associated with the user's phone number may improve the reliability of the authentication context. Any revised data or circumstances based upon this phone call are fed into the trust engine 110 for use in consideration of the authentication request.

Additionally, in STEP 1820 the trust engine 110 may provide an opportunity for the user to purchase insurance, effectively buying a more confident authentication. The operator of the trust engine 110 may, at times, only want to make such an option available if the confidence level of the authentication is above a certain threshold to begin with. In effect, this user side insurance is a way for the trust engine 110 to vouch for the user when the authentication meets the normal required trust level of the trust engine 110 for authentication, but does not meet the required trust level of the vendor for this transaction. In this way, the user may still successfully authenticate to a very high level as may be required by the vendor, even though he only has authentication instances which produce confidence sufficient for the trust engine 110.

This function of the trust engine 110 allows the trust engine 110 to vouch for someone who is authenticated to the satisfaction of the trust engine 110, but not of the vendor. This is analogous to the function performed by a

notary in adding his signature to a document in order to indicate to someone reading the document at a later time that the person whose signature appears on the document is in fact the person who signed it. The signature of the notary testifies to the act of signing by the user. In the same way, the trust engine is providing an indication that the person transacting is who they say they are.

However, because the trust engine 110 is artificially boosting the level of confidence provided by the user, there is a greater risk to the trust engine 110 operator, since the user is not actually meeting the required trust level of the vendor. The cost of the insurance is designed to offset the risk of a false positive authentication to the trust engine 110 (who may be effectively notarizing the authentications of the user). The user pays the trust engine 110 operator to take the risk of authenticating to a higher level of confidence than has actually been provided.

Because such an insurance system allows someone to effectively buy a higher confidence rating from the trust engine 110, both vendors and users may wish to prevent the use of user side insurance in certain transactions. Vendors may wish to limit positive authentications to circumstances where they know that actual authentication data supports the degree of confidence which they require and so may indicate to the trust engine 110 that user side insurance is not to be allowed. Similarly, to protect his online identity, a user may wish to prevent the use of user side insurance on his account, or may wish to limit its use to situations where the authentication confidence level without the insurance is higher than a certain limit. This may be used as a security measure to prevent someone from overhearing a password or stealing a smart card and using them to falsely authenticate to a low level of confidence, and then purchasing insurance to produce a very high level of (false) confidence. These factors may be evaluated in determining whether user side insurance is allowed.

If user purchases insurance in STEP 1840, then the authentication confidence level is adjusted based upon the insurance purchased in STEP 1845, and the authentication confidence level and required trust level are again compared in STEP 1730 (see FIGURE 17). The process continues from there, and may lead to either a positive authentication in STEP 1740 (see FIGURE 17), or back into the trust arbitrage process in STEP 1750 for either further arbitrage (if allowed) or a negative authentication in STEP 1810 if further arbitrage is prohibited.

In addition to sending a message to the user in STEP 1820, the transaction engine 205 may also send a message to the vendor in STEP 1830 which indicates that a pending authentication is currently below the required trust level. The message may also offer various options on how to proceed to the vendor. One of these options is to simply inform the vendor of what the current authentication confidence level is and ask if the vendor wishes to maintain their current unfulfilled required trust level. This may be beneficial because in some cases, the vendor may have independent means for authenticating the transaction or may have been using a default set of requirements which generally result in a higher required level being initially specified than is actually needed for the particular transaction at hand.

For instance, it may be standard practice that all incoming purchase order transactions with the vendor are expected to meet a 98% trust level. However, if an order was recently discussed by phone between the vendor and a long-standing customer, and immediately thereafter the transaction is authenticated, but only to a 93% confidence

level, the vendor may wish to simply lower the acceptance threshold for this transaction, because the phone call effectively provides additional authentication to the vendor. In certain circumstances, the vendor may be willing to lower their required trust level, but not all the way to the level of the current authentication confidence. For instance, the vendor in the above example might consider that the phone call prior to the order might merit a 4% reduction in the degree of trust needed; however, this is still greater than the 93% confidence produced by the user.

If the vendor does adjust their required trust level in STEP 1835, then the authentication confidence level produced by the authentication and the required trust level are compared in STEP 1730 (see FIGURE 17). If the confidence level now exceeds the required trust level, a positive authentication may be generated in the transaction engine 205 in STEP 1740 (see FIGURE 17). If not, further arbitrage may be attempted as discussed above if it is permitted.

In addition to requesting an adjustment to the required trust level, the transaction engine 205 may also offer vendor side insurance to the vendor requesting the authentication. This insurance serves a similar purpose to that described above for the user side insurance. Here, however, rather than the cost corresponding to the risk being taken by the trust engine 110 in authenticating above the actual authentication confidence level produced, the cost of the insurance corresponds to the risk being taken by the vendor in accepting a lower trust level in the authentication.

Instead of just lowering their actual required trust level, the vendor has the option of purchasing insurance to protect itself from the additional risk associated with a lower level of trust in the authentication of the user. As described above, it may be advantageous for the vendor to only consider purchasing such insurance to cover the trust gap in conditions where the existing authentication is already above a certain threshold.

The availability of such vendor side insurance allows the vendor the option to either: lower his trust requirement directly at no additional cost to himself, bearing the risk of a false authentication himself (based on the lower trust level required); or, buying insurance for the trust gap between the authentication confidence level and his requirement, with the trust engine 110 operator bearing the risk of the lower confidence level which has been provided. By purchasing the insurance, the vendor effectively keeps his high trust level requirement, because the risk of a false authentication is shifted to the trust engine 110 operator.

If the vendor purchases insurance in STEP 1840, the authentication confidence level and required trust levels are compared in STEP 1730 (see FIGURE 17), and the process continues as described above.

Note that it is also possible that both the user and the vendor respond to messages from the trust engine 110. Those of skill in the art will recognize that there are multiple ways in which such situations can be handled. One advantageous mode of handling the possibility of multiple responses is simply to treat the responses in a first-come, first-served manner. For example, if the vendor responds with a lowered required trust level and immediately thereafter the user also purchases insurance to raise his authentication level, the authentication is first reevaluated based upon the lowered trust requirement from the vendor. If the authentication is now positive, the user's insurance purchase is ignored. In another advantageous mode of operation, the user might only be charged for the level of

insurance required to meet the new, lowered trust requirement of the vendor (if a trust gap remained even with the lowered vendor trust requirement).

If no response from either party is received during the trust arbitrage process at STEP 1850 within the time limit set for the authentication, the arbitrage is reevaluated in STEP 1805. This effectively begins the arbitrage process again. If the time limit was final or other circumstances prevent further arbitrage in STEP 1805, a negative authentication is generated by the transaction engine 205 in STEP 1810 and returned to the vendor in STEP 1055 (see FIGURE 10). If not, new messages may be sent to the user and vendor, and the process may be repeated as desired.

Note that for certain types of transactions, for instance, digitally signing documents which are not part of a transaction, there may not necessarily be a vendor or other third party; therefore the transaction is primarily between the user and the trust engine 110. In circumstances such as these, the trust engine 110 will have its own required trust level which must be satisfied in order to generate a positive authentication. However, in such circumstances, it will often not be desirable for the trust engine 110 to offer insurance to the user in order for him to raise the confidence of his own signature.

The process described above and shown in FIGURES 16 - 18 may be carried out using various communications modes as described above with reference to the trust engine 110. For instance, the messages may be web-based and sent using SSL connections between the trust engine 110 and applets downloaded in real time to browsers running on the user or vendor systems. In an alternate mode of operation, certain dedicated applications may be in use by the user and vendor which facilitate such arbitrage and insurance transactions. In another alternate mode of operation, secure email operations may be used to mediate the arbitrage described above, thereby allowing deferred evaluations and batch processing of authentications. Those of skill in the art will recognize that different communications modes may be used as are appropriate for the circumstances and authentication requirements of the vendor.

The following description with reference to FIGURE 19 describes a sample transaction which integrates the various aspects of the present invention as described above. This example illustrates the overall process between a user and a vendor as mediates by the trust engine 110. Although the various steps and components as described in detail above may be used to carry out the following transaction, the process illustrated focuses on the interaction between the trust engine 110, user and vendor.

The transaction begins when the user, while viewing web pages online, fills out an order form on the web site of the vendor in STEP 1900. The user wishes to submit this order form to the vendor, signed with his digital signature. In order to do this, the user submits the order form with his request for a signature to the trust engine 110 in STEP 1905. The user will also provide authentication data which will be used as described above to authenticate his identity.

In STEP 1910 the authentication data is compared to the enrollment data by the trust engine 110 as discussed above, and if a positive authentication is produced, the hash of the order form, signed with the private key of the user, is forwarded to the vendor along with the order form itself.

The vendor receives the signed form in STEP 1915, and then the vendor will generate an invoice or other contract related to the purchase to be made in STEP 1920. This contract is sent back to the user with a request for a signature in STEP 1925. The vendor also sends an authentication request for this contract transaction to the trust engine 110 in STEP 1930 including a hash of the contract which will be signed by both parties. To allow the contract to be digitally signed by both parties, the vendor also includes authentication data for itself so that the vendor's signature upon the contract can later be verified if necessary.

As discussed above, the trust engine 110 then verifies the authentication data provided by the vendor to confirm the vendor's identity, and if the data produces a positive authentication in STEP 1935, continues with STEP 1955 when the data is received from the user. If the vendor's authentication data does not match the enrollment data of the vendor to the desired degree, a message is returned to the vendor requesting further authentication. Trust arbitrage may be performed here if necessary, as described above, in order for the vendor to successfully authenticate itself to the trust engine 110.

When the user receives the contract in STEP 1940, he reviews it, generates authentication data to sign it if it is acceptable in STEP 1945, and then sends a hash of the contract and his authentication data to the trust engine 110 in STEP 1950. The trust engine 110 verifies the authentication data in STEP 1955 and if the authentication is good, proceeds to process the contract as described below. As discussed above with reference to FIGURES 17 and 18, trust arbitrage may be performed as appropriate to close any trust gap which exists between the authentication confidence level and the required authentication level for the transaction.

The trust engine 110 signs the hash of the contract with the user's private key, and sends this signed hash to the vendor in STEP 1960, signing the complete message on its own behalf, *i.e.* including a hash of the complete message (including the user's signature) encrypted with the private key 510 of the trust engine 110. This message is received by the vendor in STEP 1965. The message represents a signed contract (hash of contract encrypted using user's private key) and a receipt from the trust engine 110 (the hash of the message including the signed contract, encrypted using the trust engine 110's private key).

The trust engine 110 similarly prepares a hash of the contract with the vendor's private key in STEP 1970, and forwards this to the user, signed by the trust engine 110. In this way, the user also receives a copy of the contract, signed by the vendor, as well as a receipt, signed by the trust engine 110, for delivery of the signed contract in STEP 1975.

In addition to the foregoing, an additional aspect of the invention provides a cryptographic Service Provider Module (SPM) which may be available to a client side application as a means to access functions provided by the trust engine 110 described above. One advantageous way to provide such a service is for the cryptographic SPM is to mediate communications between a third party Application Programming Interface (API) and a trust engine 110 which is accessible via a network or other remote connection. A sample cryptographic SPM is described below with reference to FIGURE 20.

For example, on a typical system, a number of API's are available to programmers. Each API provides a set of function calls which may be made by an application 2000 running upon the system. Examples of API's which provide programming interfaces suitable for cryptographic functions, authentication functions, and other security function include the Cryptographic API (CAPI) 2010 provided by Microsoft with its Windows operating systems, and the Common Data Security Architecture (CDSA), sponsored by IBM, Intel and other members of the Open Group. CAPI will be used as an exemplary security API in the discussion that follows. However, the cryptographic SPM described could be used with CDSA or other security API's as are known in the art.

This API is used by a user system 105 or vendor system 120 when a call is made for a cryptographic function. Included among these functions may be requests associated with performing various cryptographic operations, such as encrypting a document with a particular key, signing a document, requesting a digital certificate, verifying a signature upon a signed document, and such other cryptographic functions as are described above or known to those of skill in the art.

Such cryptographic functions are normally performed locally to the system upon which CAPI 2010 is located. This is because generally the functions called require the use of either resources of the local user system 105, such as a fingerprint reader, or software functions which are programmed using libraries which are executed on the local machine. Access to these local resources is normally provided by one or more Service Provider Modules (SPM's) 2015, 2020 as referred to above which provide resources with which the cryptographic functions are carried out. Such SPM's may include software libraries 2015 to perform encrypting or decrypting operations, or drivers and applications 2020 which are capable of accessing specialized hardware 2025, such as biometric scanning devices. In much the way that CAPI 2010 provides functions which may be used by an application 2000 of the system 105, the SPM's 2015, 2020 provide CAPI with access to the lower level functions and resources associated with the available services upon the system.

In accordance with the invention, it is possible to provide a cryptographic SPM 2030 which is capable of accessing the cryptographic functions provided by the trust engine 110 and making these functions available to an application 2000 through CAPI 2010. Unlike embodiments where CAPI 2010 is only able to access resources which are locally available through SPM's 2015, 2020, a cryptographic SPM 2030 as described herein would be able to submit requests for cryptographic operations to a remotely-located, network-accessible trust engine 110 in order to perform the operations desired.

For instance, if an application 2000 has a need for a cryptographic operation, such as signing a document, the application 2000 makes a function call to the appropriate CAPI 2010 function. CAPI 2010 in turn will execute this function, making use of the resources which are made available to it by the SPM's 2015, 2020 and the cryptographic SPM 2030. In the case of a digital signature function, the cryptographic SPM 2030 will generate an appropriate request which will be sent to the trust engine 110 across the communication link 125.

The operations which occur between the cryptographic SPM 2030 and the trust engine 110 are the same operations that would be possible between any other system and the trust engine 110. However, these functions are

effectively made available to a user system 105 through CAPI 2010 such that they appear to be locally available upon the user system 105 itself. However, unlike ordinary SPM's 2015, 2020, the functions are being carried out on the remote trust engine 110 and the results relayed to the cryptographic SPM 2030 in response to appropriate requests across the communication link 125.

This cryptographic SPM 2030 makes a number of operations available to the user system 105 or a vendor system 120 which might not otherwise be available. These functions include without limitation: encryption and decryption of documents; issuance of digital certificates; digital signing of documents; verification of digital signatures; and such other operations as will be apparent to those of skill in the art.

Additionally, other combinations, admissions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein. Accordingly, the present invention is not intended to be limited by the reaction of the preferred embodiments but is to be defined by a reference to the appended claims.

WHAT IS CLAIMED IS:

1. A remotely accessible secure cryptographic system for storing a plurality of private cryptographic keys to be associated with a plurality of users, wherein the cryptographic system associates each of the plurality of users with one or more different keys from the plurality of private cryptographic keys and performs cryptographic functions for each user using the associated one or more different keys without releasing the plurality of private cryptographic keys to the users, the cryptographic system comprising:

a depository system having at least one server which stores a plurality of private cryptographic keys and a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of multiple users and each of the multiple users is associated with one or more different keys from the plurality of private cryptographic keys;

an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result;

a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using the associated one or more different keys received from the depository system; and

a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

2. A remotely accessible secure cryptographic system, comprising:

a depository system having at least one server which stores at least one private key and a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of multiple users;

an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result;

a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using at least said private key received from the depository system; and

a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

3. The cryptographic system of Claim 2, wherein the depository system further comprises a plurality of data storage facilities, each data storage facility having at least one server storing a substantially randomized portion of the private key and a substantially randomized portion of the plurality of enrollment authentication data.

4. The cryptographic system of Claim 3, wherein each substantially randomized portion is individually undecipherable.

5. The cryptographic system of Claim 2, wherein the enrollment authentication data includes biometric data.

6. The cryptographic system of Claim 5, wherein the biometric data includes finger print patterns.

7. The cryptographic system of Claim 2, wherein the at least one private key corresponds to the secure cryptographic system.

8. The cryptographic system of Claim 2, wherein the at least one private key corresponds to the one of the multiple users.

9. The trust engine of Claim 2, wherein the cryptographic functions comprise one of digital signing, encryption, and decryption.

10. A method of facilitating cryptographic functions, the method comprising:

associating a user from multiple users with one or more keys from a plurality of private cryptographic keys stored on a secure server;

receiving authentication data from the user;

comparing the authentication data to authentication data corresponding to the user, thereby verifying the identity of the user; and

utilizing the one or more keys to perform cryptographic functions without releasing the one or more keys to the user.

11. The method of Claim 10, wherein the authentication data corresponding to the user was acquired prior to the step of receiving authentication data from the user.

12. The method of Claim 10, further comprising receiving a hash of a message or document.

13. The method of Claim 12, further comprising archiving the hash.

14. An authentication system for uniquely identifying a user through secure storage of the user's enrollment authentication data, the authentication system comprising:

a plurality of data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores one of portions of enrollment authentication data; and

an authentication engine which communicates with the plurality of data storage facilities and comprises

a data splitting module which operates on the enrollment authentication data to create portions,

a data assembling module which processes the portions from at least two of the data storage facilities to assemble the enrollment authentication data, and

a data comparator module which receives current authentication data from a user and compares the current authentication data with the assembled enrollment authentication data to determine whether the user has been uniquely identified.

15. The authentication system of Claim 14, wherein the portions are not individually decipherable.

16. The authentication system of Claim 14, wherein the each data storage facility is logically separated from any other data storage facility.

17. The authentication system of Claim 14, wherein the each data storage facility is physically separated from any other data storage facility.

18. The authentication system of Claim 14, further comprising a cryptographic engine which, upon the unique identification of the user by the authentication engine, provides cryptographic functionality to the user.

19. The authentication system of Claim 14, wherein the plurality of data storage facilities comprises at least one secure server.

20. The authentication system of Claim 14, wherein unique identification of the user by the authentication engine provides the user authorization to gain access to or to operate one or more systems.

21. The authentication system of Claim 20, wherein the one or more systems include one or more electronic devices.

22. The authentication system of Claim 20, wherein the one or more systems include one or more computer software systems.

23. The authentication system of Claim 20, wherein the one or more systems include one or more consumer electronics.

24. The authentication system of Claim 23, wherein the one or more consumer electronics includes a cellular phone.

25. The authentication system of Claim 20, wherein the one or more systems include one or more cryptographic systems.

26. The authentication system of Claim 20, wherein the one or more systems include one or more physical locations.

27. The authentication system of Claim 14, wherein at least one of the data storage facilities stores at least some of sensitive data, wherein the at least one of the data storage facilities serves the sensitive data when the authentication engine indicates that the user has been uniquely identified.

28. The authentication system of Claim 14, further comprising a data vault which stores sensitive data, wherein the data vault serves the sensitive data when the authentication engine indicates that the user has been uniquely identified.

29. The authentication system of Claim 14, wherein the authentication system outputs an indication of whether the user has been uniquely identified.

30. A cryptographic system, comprising:

a plurality of data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores one of portions of cryptographic keys; and

a cryptographic engine which communicates with the plurality of data storage facilities and comprises

a data splitting module which operate on the cryptographic keys to create portions,

a data assembling module which processes the portions from at least two of the data storage facilities to assemble the cryptographic keys, and

a cryptographic handling module which receives the assembled cryptographic keys and performs cryptographic functions therewith.

31. The cryptographic system of Claim 30, wherein the portions are not individually decipherable.

32. The cryptographic system of Claim 30, wherein the each data storage facility is logically separated from any other data storage facility.

33. The cryptographic system of Claim 30, wherein the each data storage facility is physically separated from any other data storage facility.

34. The cryptographic system of Claim 30, further comprising an authentication engine which, before the cryptographic functionality may be employed on behalf of a user, uniquely identifies the user.

35. The cryptographic system of Claim 30, wherein the plurality of data storage facilities comprises at least one secure server.

36. A method of storing authentication data in geographically remote secure data storage facilities thereby protecting the authentication data against comprise of any individual data storage facility, the method comprising:

receiving authentication data at a trust engine;

combining at the trust engine the authentication data with a first substantially random value to form a first combined value;

combining the authentication data with a second substantially random value to form a second combined value;

creating a first pairing of the first substantially random value with the second combined value;

creating a second pairing of the first substantially random value with the second substantially random value;

storing the first pairing in a first secure data storage facility; and

storing the second pairing in a second secure data storage facility remote from the first secure data storage facility.

37. A method of storing authentication data comprising:

receiving authentication data;

combining the authentication data with a first set of bits to form a second set of bits;

combining the authentication data with a third set of bits to form a fourth set of bits;

creating a first pairing of the first set of bits with the third set of bits;

creating a second pairing of the first set of bits with the fourth set of bits;

storing one of the first and second pairings in a first computer accessible storage medium; and

storing the other of the first and second pairings in a second computer accessible storage medium.

38. The method of Claim 37, wherein at least one of the first and second computer accessible storage mediums comprises at least one server.

39. The method of Claim 37, wherein the first computer accessible storage medium is geographically remote from the second computer accessible storage medium.

40. The method of Claim 37, wherein the matching of one of the first and second pairings with one of the first and second computer accessible storage mediums is substantially random.

41. The method of Claim 37, wherein at least one of the first and third sets of bits are substantially random.

42. The method of Claim 37, wherein at least one of the first and third sets of bits comprises a bit length equal to a bit length of the sensitive data.

43. The method of Claim 37, wherein both the first and second pairings are needed to reassemble the data.

44. The method of Claim 37, further comprising:

creating a third pairing of the second set of bits with the third set of bits;

creating a fourth pairing of the second set of bits with the fourth set of bits;

storing one of the third and fourth pairings in a third computer accessible storage medium; and

storing the other of the third and fourth pairings in a fourth computer accessible storage medium.

45. A method of storing cryptographic data in geographically remote secure data storage facilities thereby protecting the cryptographic data against comprise of any individual data storage facility, the method comprising:

receiving cryptographic data at a trust engine;

combining at the trust engine the cryptographic data with a first substantially random value to form a first combined value;

combining the cryptographic data with a second substantially random value to form a second combined value;

creating a first pairing of the first substantially random value with the second combined value;

creating a second pairing of the first substantially random value with the second substantially random value;

storing the first pairing in a first secure data storage facility; and

storing the second pairing in a secure second data storage facility remote from the first secure data storage facility.

46. A method of storing cryptographic data comprising:

receiving authentication data;

combining the cryptographic data with a first set of bits to form a second set of bits;

combining the cryptographic data with a third set of bits to form a fourth set of bits;

creating a first pairing of the first set of bits with the third set of bits;

creating a second pairing of the first set of bits with the fourth set of bits;

storing one of the first and second pairings in a first computer accessible storage medium; and

storing the other of the first and second pairings in a second computer accessible storage medium.

47. The method of Claim 46, wherein at least one of the first and second computer accessible storage mediums comprises at least one server.

48. The method of Claim 46, wherein the first computer accessible storage medium is geographically remote from the second computer accessible storage medium.

49. The method of Claim 46, wherein the matching of one of the first and second pairings with one of the first and second computer accessible storage mediums is substantially random.

50. The method of Claim 46, wherein at least one of the first and third sets of bits are substantially random.

51. The method of Claim 46, wherein at least one of the first and third sets of bits comprises a bit length equal to a bit length of the sensitive data.

52. The method of Claim 46, wherein both the first and second pairings are needed to reassemble the cryptographic data.

53. The method of Claim 46, further comprising:

creating a third pairing of the second set of bits with the third set of bits;

creating a fourth pairing of the second set of bits with the fourth set of bits;

storing one of the third and fourth pairings in a third computer accessible storage medium; and

storing the other of the third and fourth pairings in a fourth computer accessible storage medium.

54. A method of handling sensitive data in a cryptographic system, wherein the sensitive data exists in a useable form only during actions employing the sensitive data, the method comprising:

receiving in a software module, substantially randomized sensitive data from a first computer accessible storage medium;

receiving in the software module, substantially randomized data from a second computer accessible storage medium,

processing the substantially randomized sensitive data and the substantially randomized data in the software module to assemble the sensitive data; and

employing the sensitive data in a software engine to perform an action, wherein the action includes one of authenticating a user and performing a cryptographic function.

55. The method of Claim 54, further comprising destroying the sensitive data after completion of the action.

56. The method of Claim 54, wherein the sensitive data includes one of user biometric data and cryptographic key data.

57. The method of Claim 54, wherein at least one of the first and second computer accessible storage mediums comprise a secure server.

58. The method of Claim 54, wherein the software module comprises a data assembling module and the software engine comprises one of an authentication engine and a cryptographic engine.

59. A secure authentication system, comprising:

a plurality of authentication engines, wherein each authentication engine receives enrollment authentication data designed to uniquely identify a user to a degree of certainty, each authentication engine receives current authentication data to compare to the enrollment authentication data, and wherein each authentication engine determines an authentication result; and

a redundancy system which receives the authentication result of at least two of the authentication engines and determines whether the user has been uniquely identified.

60. The secure authentication system of Claim 59, wherein the redundancy system determines whether the user has been uniquely identified by following the majority of the authentication results.

61. The secure authentication system of Claim 59, wherein the redundancy system determines whether the user has been uniquely identified by requiring the authentication results to be unanimously positive before issuing a positive identification.

62. The secure authentication system of Claim 59, wherein the redundancy system includes a plurality of redundancy modules, and the secure authentication system further comprises:

a plurality of geographically remote trust engines, each trust engine having one of the plurality of authentication engines and one of the redundancy modules,

wherein the redundancy module for at least one of the plurality of trust engines determines whether the user has been uniquely identified using the authentication results from ones of the authentication engines associated with the other trust engines and without using the authentication results from the at least one trust engine.

63. The secure authentication system of Claim 62, wherein each of the plurality of trust engines includes a depository having a computer accessible storage medium which stores a substantially randomized portion of the enrollment authentication data and wherein each depository forwards the substantially randomized portion of the enrollment authentication data to the plurality of authentication engines.

64. The secure authentication system of Claim 62, wherein the determination of whether the user has been uniquely identified corresponds to the one of the redundancy modules to first determine a result.

65. A trust engine system for facilitating authentication of a user, the trust engine system comprising:

a first trust engine comprising a first depository, wherein the first depository includes a computer accessible storage medium which stores portions of enrollment authentication data;

a second trust engine located at a different geographic location than the first trust engine and comprising

a second depository having a computer accessible storage medium which stores portions of enrollment authentication data,

an authentication engine communicating with the first and second depositories and which assembles at least two portions of enrollment authentication data into a usable form, and

a transaction engine communicating with the first and second depositories and the authentication engine,

wherein when the second trust engine is determined to be available to execute a transaction, the transaction engine receives authentication data from a user and forwards a request for the portions of enrollment authentication data to the first and second depositories, and wherein the authentication engine receives the authentication data from the transaction engine and the portions of the enrollment authentication data from the first and second depositories, and determines an authentication result.

66. The trust engine system of Claim 65, wherein the determination of whether the second trust engine is available to execute the transaction includes a determination of whether the second trust engine is within geographic proximity to the user.

67. The trust engine system of Claim 65, wherein the determination of whether the second trust engine is available to execute the transaction includes a determination of whether the second trust engine is currently servicing a light system load.

68. The trust engine system of Claim 65, wherein the determination of whether the second trust engine is available to execute the transaction includes a determination of whether the second trust engine is currently scheduled for maintenance.

69. The trust engine system of Claim 65, wherein the first and second trust engines are determined to be available, and an authentication result for the trust engine system follows the first of the first and second trust engines to produce the authentication result.

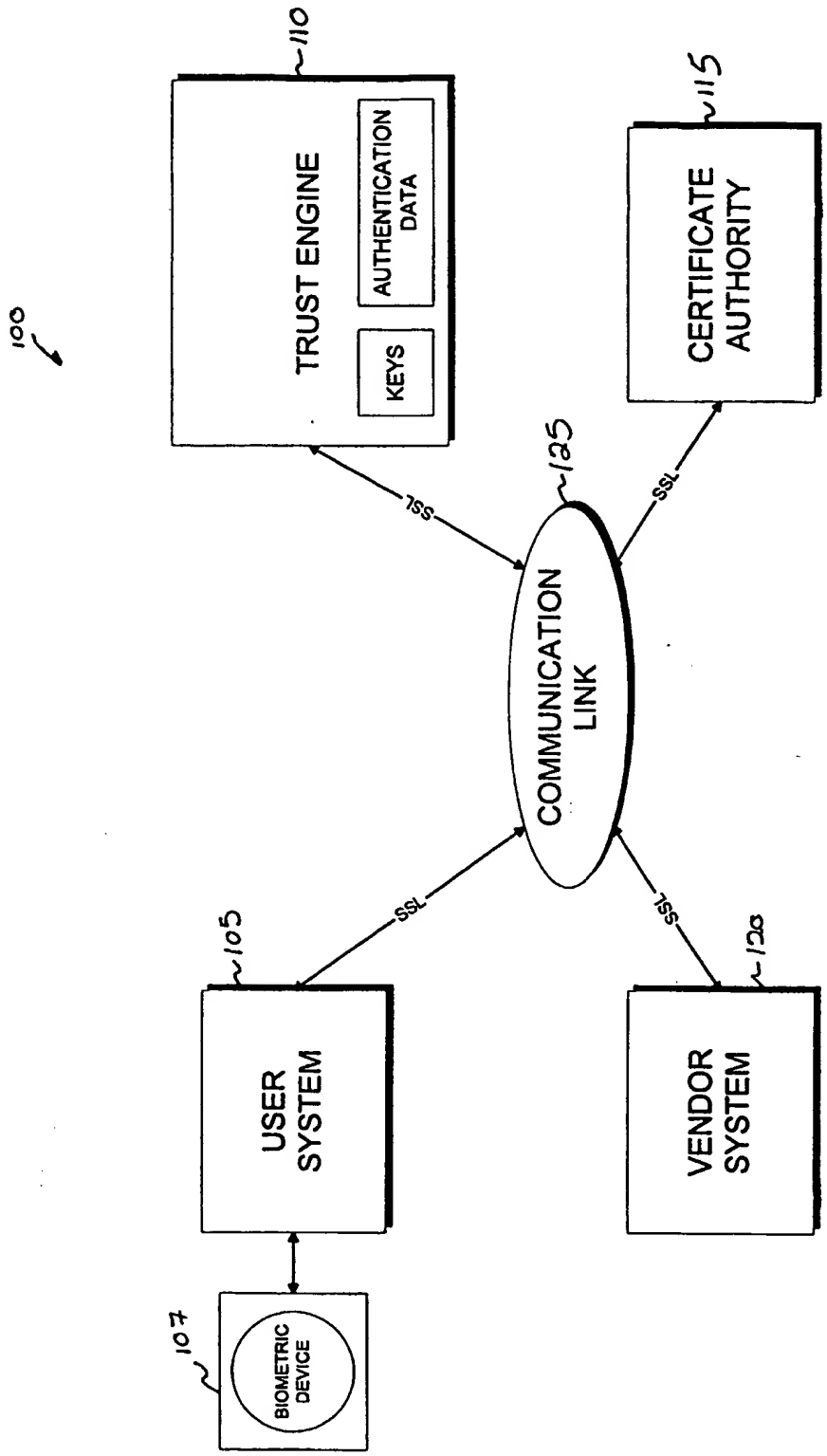

100
110
TRUST ENGINE
AUTHENTICATION DATA
KEYS
115
CERTIFICATE AUTHORITY
125
COMMUNICATION LINK
SSL
SSL
SSL
SSL
105
USER SYSTEM
120
VENDOR SYSTEM
107
BIOMETRIC DEVICE

FIG. 1

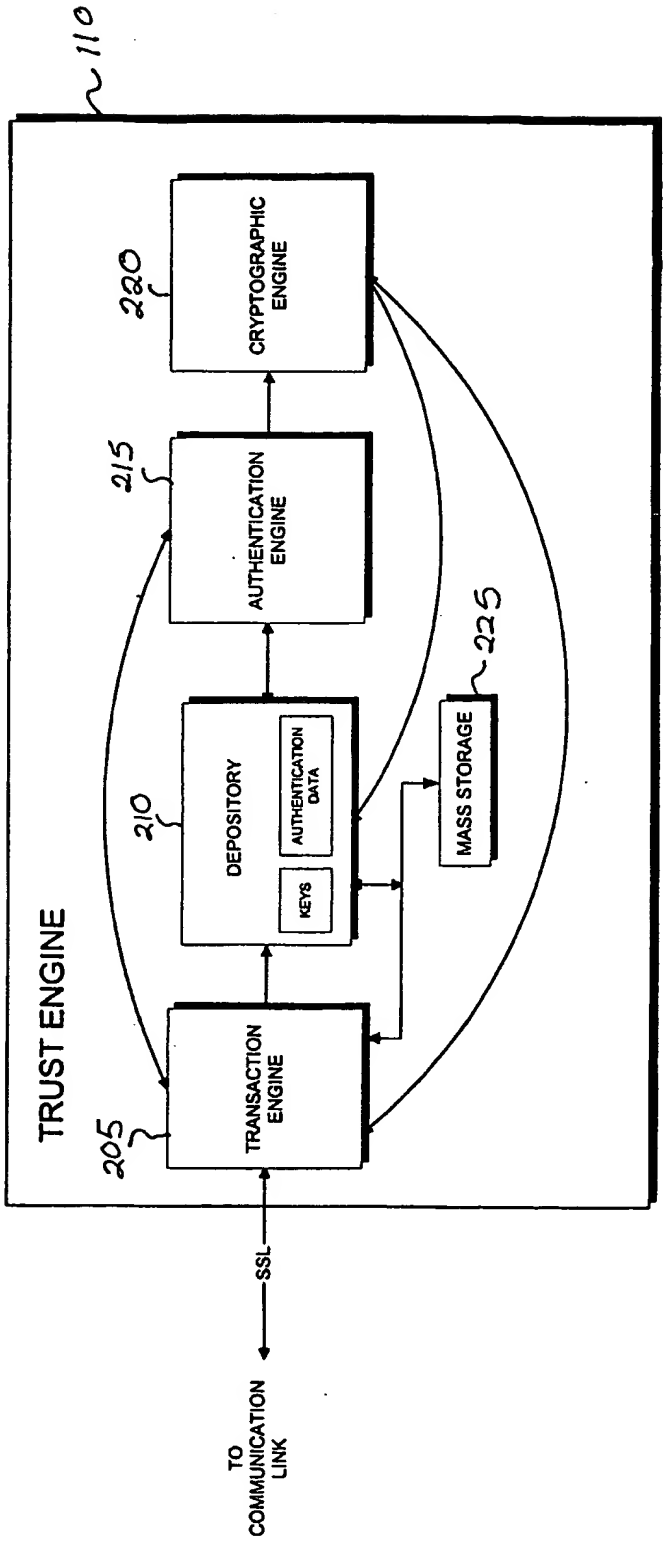

110
TRUST ENGINE
220
CRYPTOGRAPHIC ENGINE
215
AUTHENTICATION ENGINE
210
DEPOSITORY
AUTHENTICATION DATA
KEYS
225
MASS STORAGE
205
TRANSACTION ENGINE
SSL
TO COMMUNICATION LINK

FIG. 2

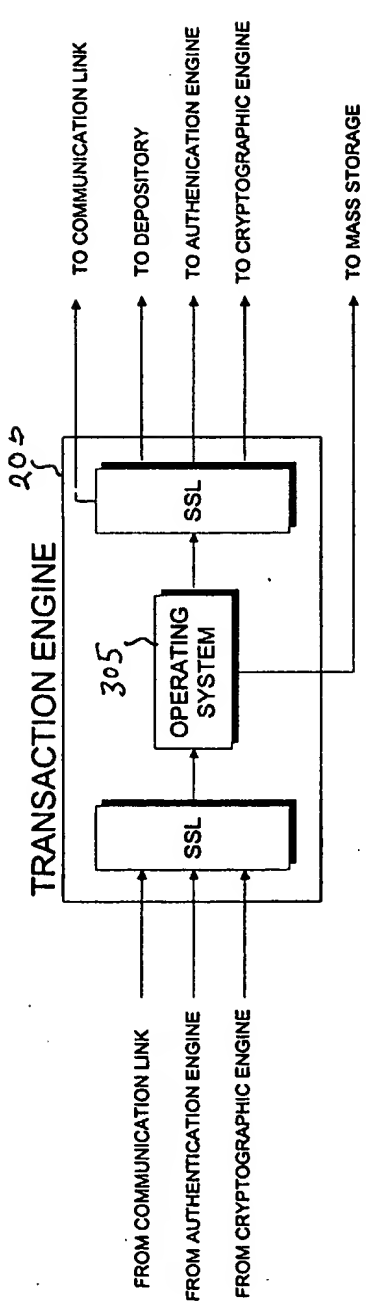

TRANSACTION ENGINE
205
SSL
OPERATING SYSTEM
305
SSL
FROM COMMUNICATION LINK
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
TO COMMUNICATION LINK
TO DEPOSITORY
TO AUTHENICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
TO MASS STORAGE

FIG. 3

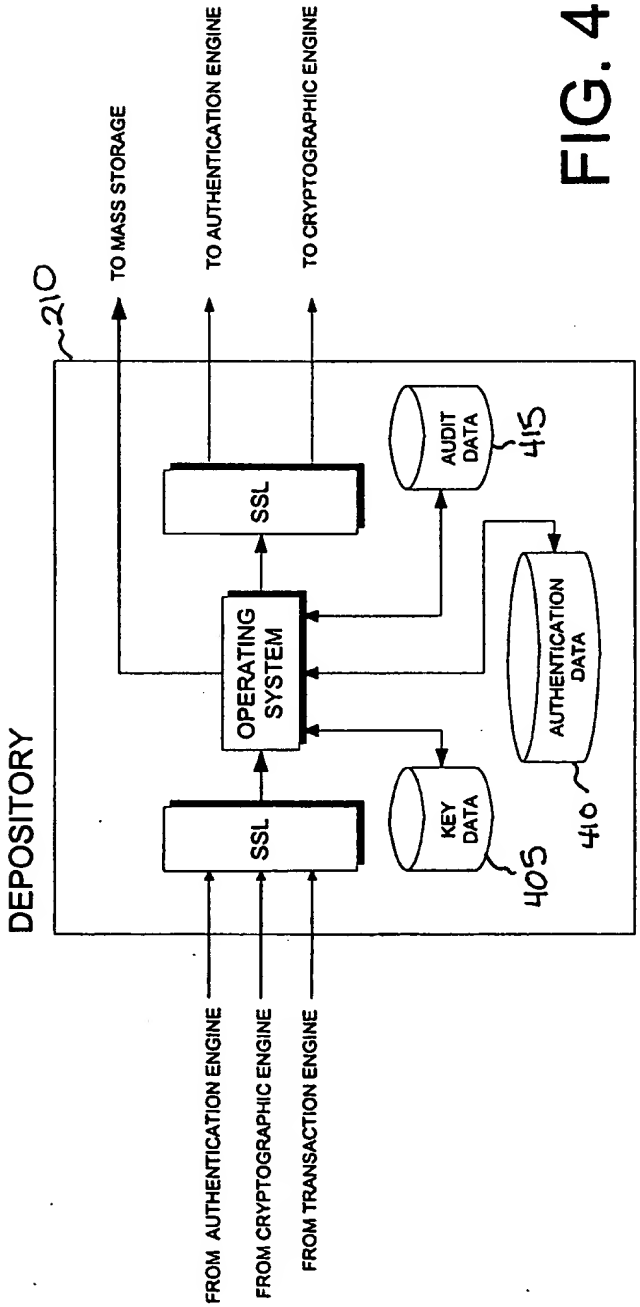

DEPOSITORY
210
SSL
OPERATING SYSTEM
SSL
KEY DATA
405
AUDIT DATA
415
AUTHENTICATION DATA
410
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
FROM TRANSACTION ENGINE
TO MASS STORAGE
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE

FIG. 4

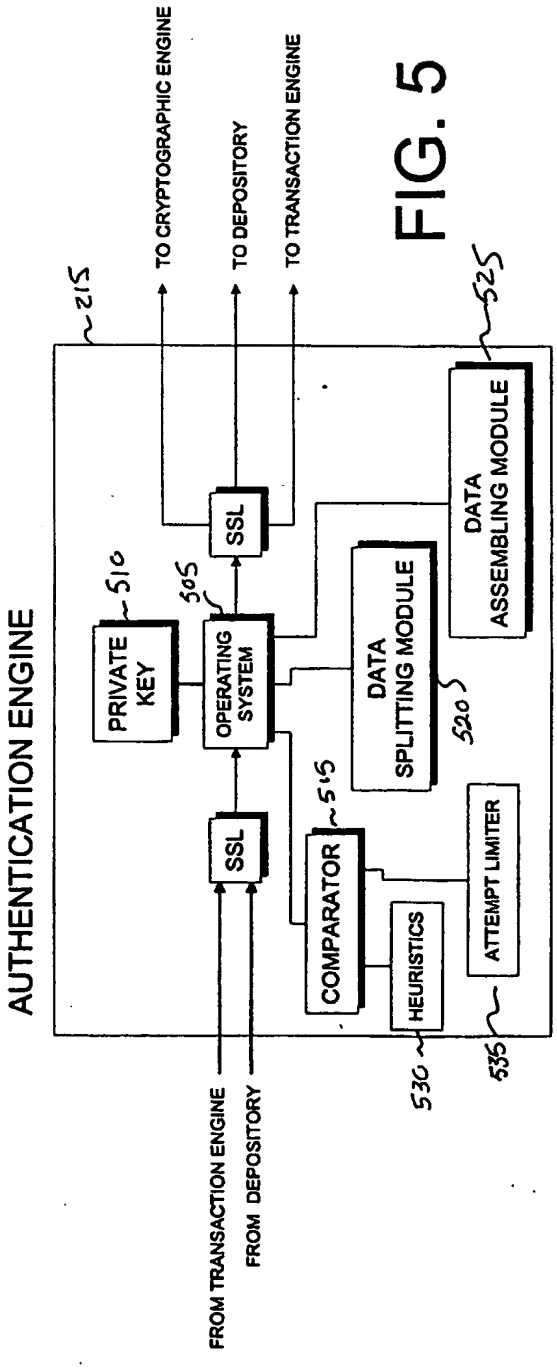

AUTHENTICATION ENGINE
215
TO CRYPTOGRAPHIC ENGINE
TO DEPOSITORY
TO TRANSACTION ENGINE
PRIVATE KEY
510
OPERATING SYSTEM
505
SSL
SSL
DATA SPLITTING MODULE
520
DATA ASSEMBLING MODULE
525
COMPARATOR
515
HEURISTICS
530
ATTEMPT LIMITER
535
FROM TRANSACTION ENGINE
FROM DEPOSITORY

FIG. 5

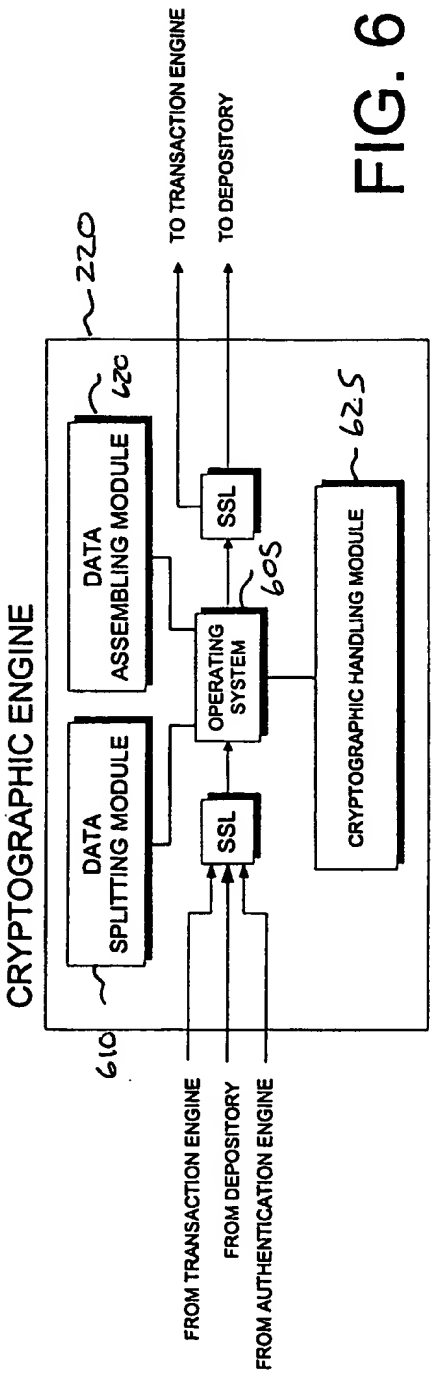

CRYPTOGRAPHIC ENGINE
220
TO TRANSACTION ENGINE
TO DEPOSITORY
DATA ASSEMBLING MODULE
620
DATA SPLITTING MODULE
610
OPERATING SYSTEM
605
SSL
SSL
CRYPTOGRAPHIC HANDLING MODULE
625
FROM TRANSACTION ENGINE
FROM DEPOSITORY
FROM AUTHENTICATION ENGINE

FIG. 6

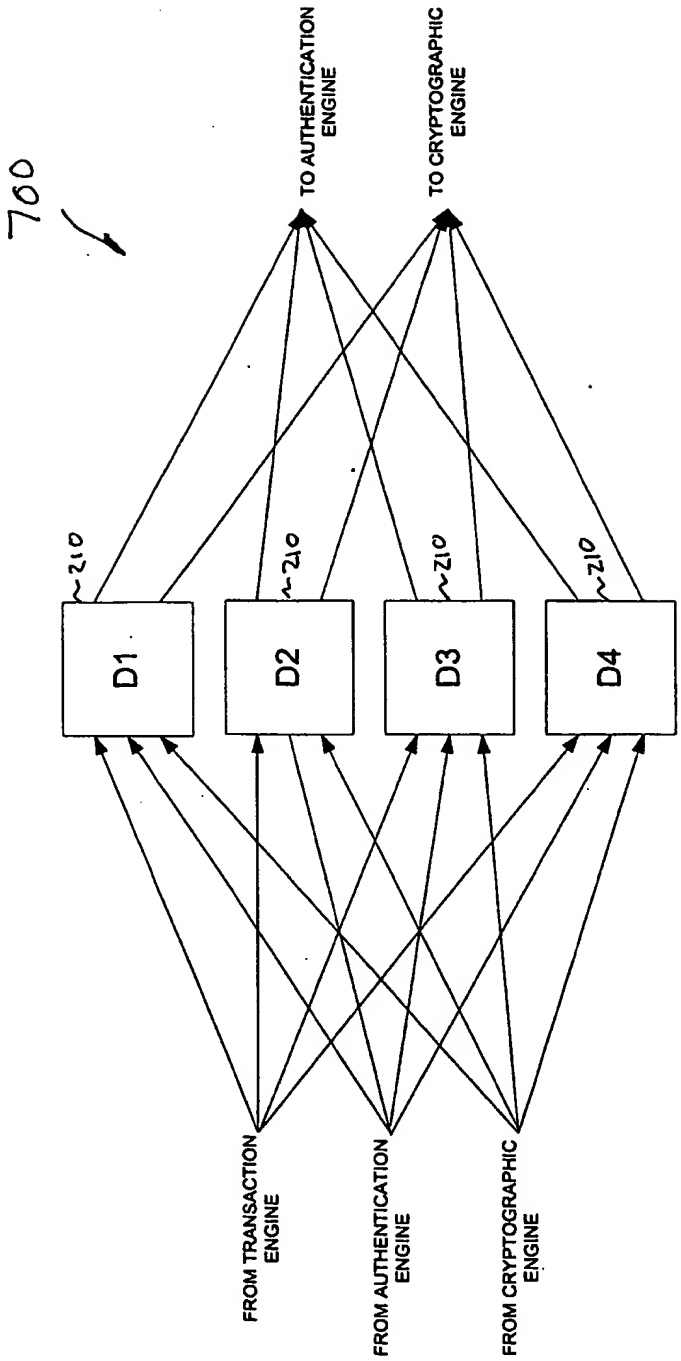

700
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
210
210
210
210
D1
D2
D3
D4
FROM TRANSACTION ENGINE
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE

FIG. 7

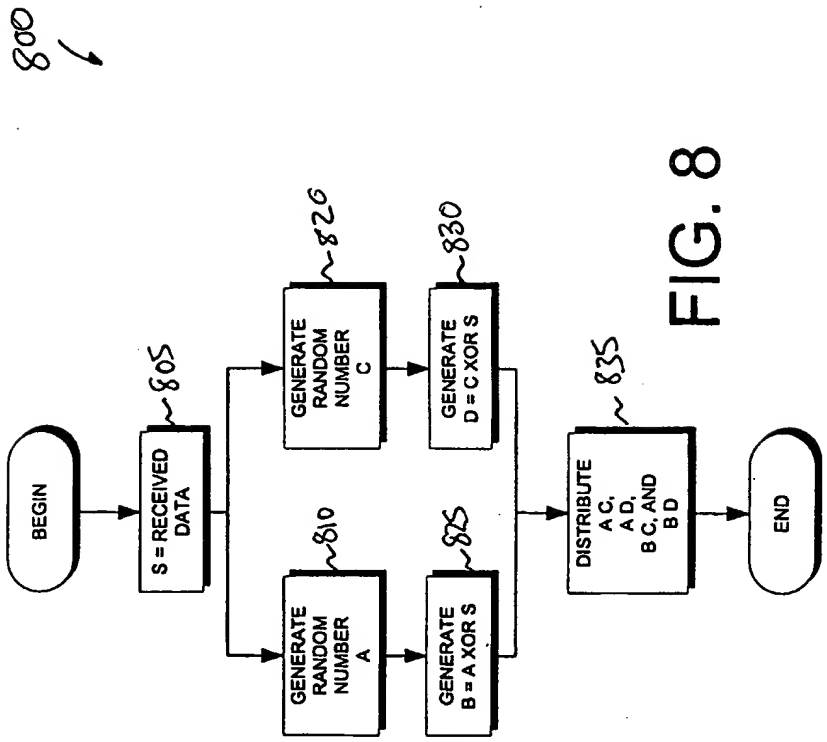

800
BEGIN
S = RECEIVED DATA
805
GENERATE RANDOM NUMBER A
810
GENERATE RANDOM NUMBER C
820
GENERATE B = A XOR S
825
GENERATE D = C XOR S
830
DISTRIBUTE A C, A D, B C, AND B D
835
END

FIG. 8

900

	ENROLLMENT DATA FLOW			
	SEND	**RECEIVE**	**SSL**	**ACTION**
905	USER	TRANSACTION ENGINE (TE)	½	TRANSMIT ENROLLMENT AUTHENTICATION DATA (B) AND THE USER ID (UID) ENCRYPTED WITH THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(UID, B))
915	TE	AE	FULL	FORWARD TRANSMISSION
920				AE DECRYPTS AND SPLITS FORWARDED DATA
925	AE	THE X^{TH} DEPOSITORY (DX)	FULL	STORE RESPECTIVE PORTION OF DATA
	WHEN DIGITAL CERTIFICATE REQUESTED			
930	AE	CRYPTOGRAPHIC ENGINE (CE)	FULL	REQUEST KEY GENERATION
935				CE GENERATES AND SPLITS KEY
945	CE	TE	FULL	TRANSMIT REQUEST FOR DIGITAL CERTIFICATE
950	TE	CERTIFICATION AUTHORITY (CA)	½	TRANSMIT REQUEST
955	CA	TE	½	TRANSMIT DIGITAL CERTIFICATE
960	TE	USER	½	TRANSMIT DIGITAL CERTIFICATE
	TE	MS	FULL	STORE DIGITAL CERTIFICATE
965	CE	DX	FULL	STORE RESPECTIVE PORTION OF KEY

FIG. 9A

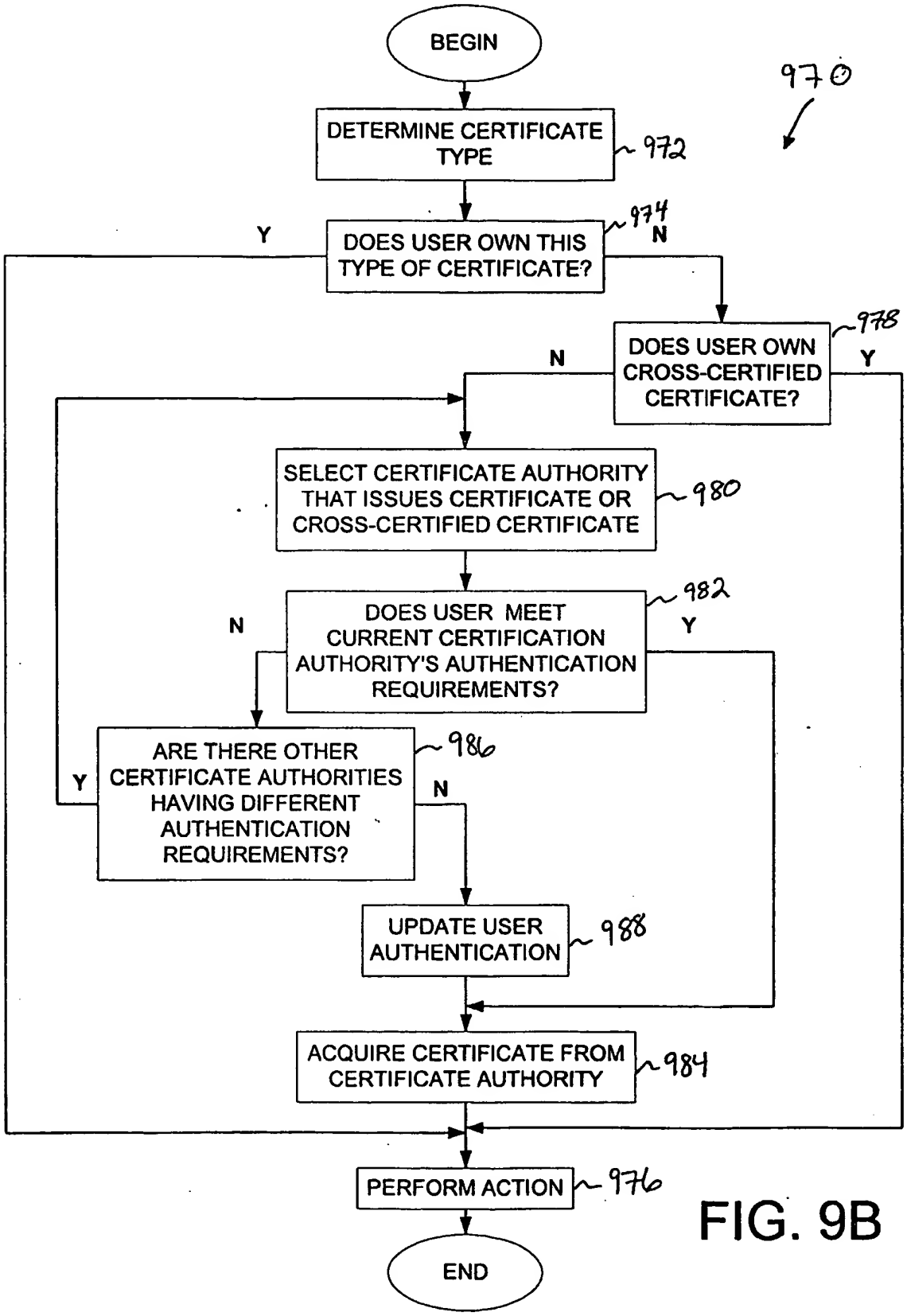

BEGIN
970
DETERMINE CERTIFICATE TYPE
972
DOES USER OWN THIS TYPE OF CERTIFICATE?
974
Y
N
DOES USER OWN CROSS-CERTIFIED CERTIFICATE?
978
N
Y
SELECT CERTIFICATE AUTHORITY THAT ISSUES CERTIFICATE OR CROSS-CERTIFIED CERTIFICATE
980
DOES USER MEET CURRENT CERTIFICATION AUTHORITY'S AUTHENTICATION REQUIREMENTS?
982
N
Y
ARE THERE OTHER CERTIFICATE AUTHORITIES HAVING DIFFERENT AUTHENTICATION REQUIREMENTS?
986
Y
N
UPDATE USER AUTHENTICATION
988
ACQUIRE CERTIFICATE FROM CERTIFICATE AUTHORITY
984
PERFORM ACTION
976
END

FIG. 9B

1000

	AUTHENTICATION DATA FLOW			
	SEND	**RECEIVE**	**SSL**	**ACTION**
005	USER	VENDOR	½	TRANSACTION OCCURS, SUCH AS SELECTING PURCHASE
1010	VENDOR	USER	½	TRANSMIT TRANSACTION ID (TID), AND AUTHENTICATION REQUEST (AR)
				AUTHENTICATION DATA (B') IS GATHERED FROM USER
1015	USER	TE	½	TRANSMIT TID AND B' WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE), AS (PUB_AE(TID, B'))
1020	TE	AE	FULL	FORWARD TRANSMISSION
				ENROLLMENT AUTHENTICATION DATA (B) IS REQUESTED AND GATHERED
1025	VENDOR	TRANSACTION ENGINE (TE)	FULL	TRANSMITS TID, AR
1030	TE	MASS STORAGE (MS)	FULL	CREATE RECORD IN DATABASE
1035	TE	THE X^{TH} DEPOSITORY (DX)	FULL	UID, TID
1040	DX	AE	FULL	TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX) AS (PUB_AE(TID, BX))
1045				AE ASSEMBLES B AND COMPARES TO B'
1050	AE	TE	FULL	TID, THE FILLED IN AR
	TE	VENDOR	FULL	TID, YES/NO
1055	TE	USER	½	TID, CONFIRMATION MESSAGE

FIG. 10

1100

SIGNING DATA FLOW			
SEND	**RECEIVE**	**SSL**	**ACTION**
USER	VENDOR	½	TRANSACTION OCCURS, SUCH AS AGREEING ON A DEAL
VENDOR	USER	½	TRANSMIT TRANSACTION IDENTIFICATION NUMBER (TID), AUTHENTICATION REQUEST (AR), AND AGREEMENT OR MESSAGE (M)
			CURRENT AUTHENTICATION DATA (B') AND A HASH OF THE MESSAGE RECEIVED BY THE USER (h(M')) IS GATHERED FROM USER
USER	TE	½	TRANSMIT TID, B', AR, AND h(M') WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(TID, B', h(M'))
TE	AE	FULL	FORWARD TRANSMISSION
			GATHER ENROLLMENT AUTHENTICATION DATA
VENDOR	TRANSACTION ENGINE (TE)	FULL	TRANSMITS UID, TID, AR, AND A HASH OF THE MESSAGE (h(M)).
TE	MASS STORAGE (MS)	FULL	CREATE RECORD IN DATABASE
TE	THE X^{TH} DEPOSITORY (DX)	FULL	UID, TID
DX	AE	FULL	TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX) AS (PUB_AE(TID, BX))
			THE ORIGINAL VENDOR MESSAGE IS TRANSMITTED TO THE AE
TE	AE	FULL	TRANSMIT h(M)
			AE ASSEMBLES B, COMPARES TO B' AND COMPARES h(M) TO h(M')
AE	CRYPTOGRAPHIC ENGINE (CE)	FULL	REQUEST FOR DIGITAL SIGNATURE AND A MESSAGE TO BE SIGNED, FOR EXAMPLE, THE HASHED MESSAGE
AE	DX	FULL	TID, SIGNING UID
DX	CE	FULL	TRANSMIT THE PORTION OF THE CRYPTOGRAPHIC KEY CORRESPONDING TO THE SIGNING PARTY
			CE ASSEMBLES KEY AND SIGNS
CE	AE	FULL	TRANSMIT THE DIGITAL SIGNATURE (S) OF SIGNING PARTY
AE	TE	FULL	TID, THE FILLED IN AR, h(M), AND S
TE	VENDOR	FULL	TID, A RECEIPT = (TID, YES/NO, AND S), AND THE DIGITAL SIGNATURE OF THE TRUSTENGINE, FOR EXAMPLE, A HASH OF THE RECEIPT ENCRYPTED WITH THE TRUSTENGINE'S PRIVATE KEY (Priv_TE(h(RECEIPT))
TE	USER	½	TID, CONFIRMATION MESSAGE

FIG. 11

1200

	ENCRYPTION/DECRYPTION DATA FLOW			
	SEND	**RECEIVE**	**SSL**	**ACTION**
	DECRYPTION			
				PERFORM AUTHENTICATION DATA PROCESS 1000, INCLUDE THE SESSION KEY (SYNC) IN THE AR, WHERE THE SYNC HAS BEEN ENCRYPTED WITH THE PUBLIC KEY OF THE USER AS PUB_USER(SNYC)
				AUTHENTICATE THE USER
1205	AE	CE	FULL	FORWARD PUB_USER(SYNC) TO CE
1210	AE	DX	FULL	UID, TID
1215	DX	CE	FULL	TRANSMIT THE TID AND THE PORTION OF THE PRIVATE KEY AS (PUB_AE(TID, KEY_USER))
1220				CE ASSEMBLES THE CRYPTOGRAPHIC KEY AND DECRYPTS THE SYNC
1225	CE	AE	FULL	TID, THE FILLED IN AR INCLUDING DECRYPTED SYNC
1230	AE	TE	FULL	FORWARD TO TE
	TE	REQUESTING APP/VENDOR	½	TID, YES/NO, SYNC
	ENCRYPTION			
1235	REQUESTING APP/VENDOR	TE	½	REQUEST FOR PUBLIC KEY OF USER
1240	TE	MS	FULL	REQUEST DIGITAL CERTIFICATE
1245	MS	TE	FULL	TRANSMIT DIGITAL CERTIFICATE
1250	TE	REQUESTING APP/VENDOR	½	TRANSMIT DIGITAL CERTIFICATE

FIG. 12

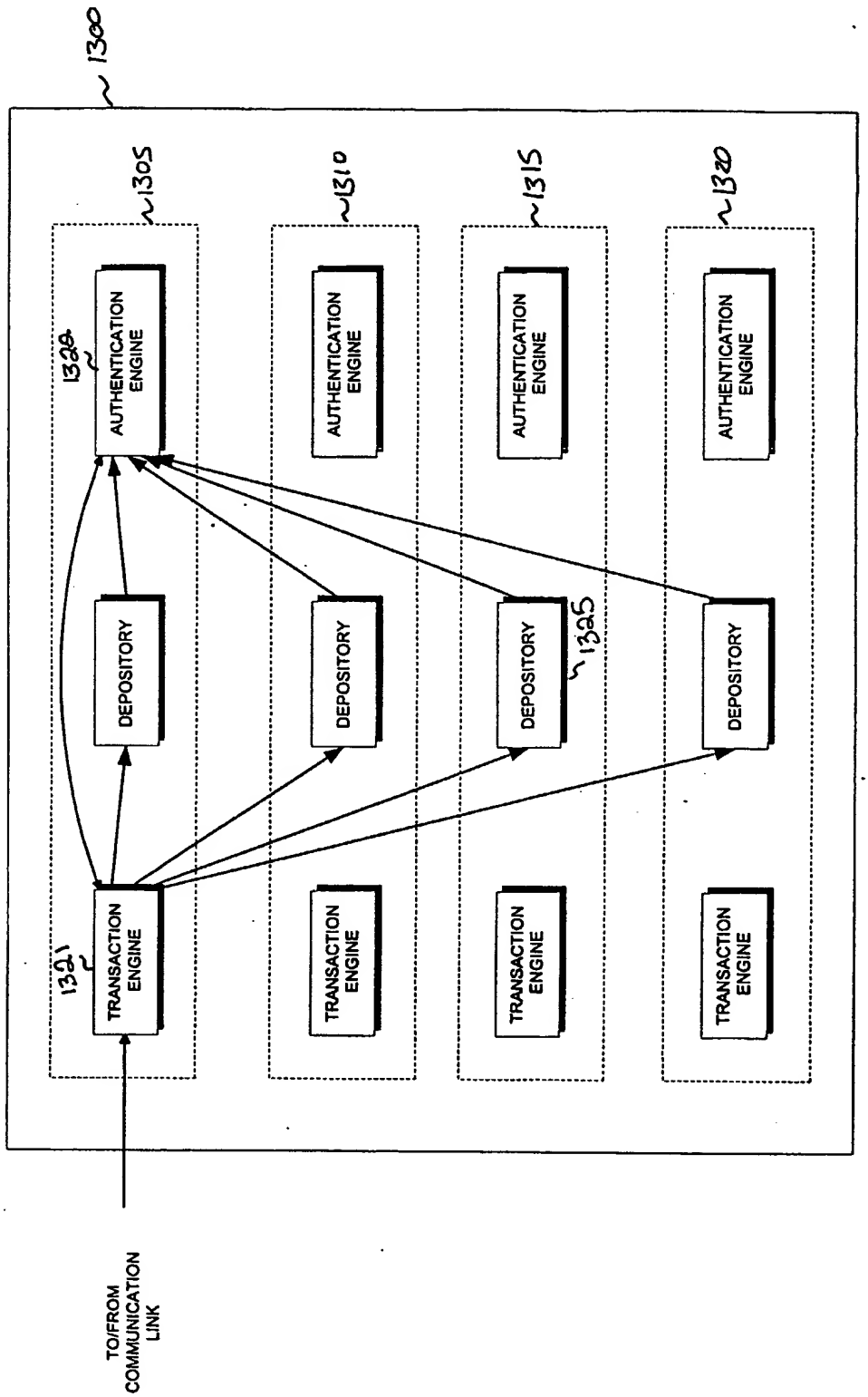

1300
1305
1310
1315
1320
1320
AUTHENTICATION ENGINE
AUTHENTICATION ENGINE
AUTHENTICATION ENGINE
AUTHENTICATION ENGINE
DEPOSITORY
DEPOSITORY
DEPOSITORY
1325
DEPOSITORY
1321
TRANSACTION ENGINE
TRANSACTION ENGINE
TRANSACTION ENGINE
TRANSACTION ENGINE
TO/FROM COMMUNICATION LINK

FIG. 13

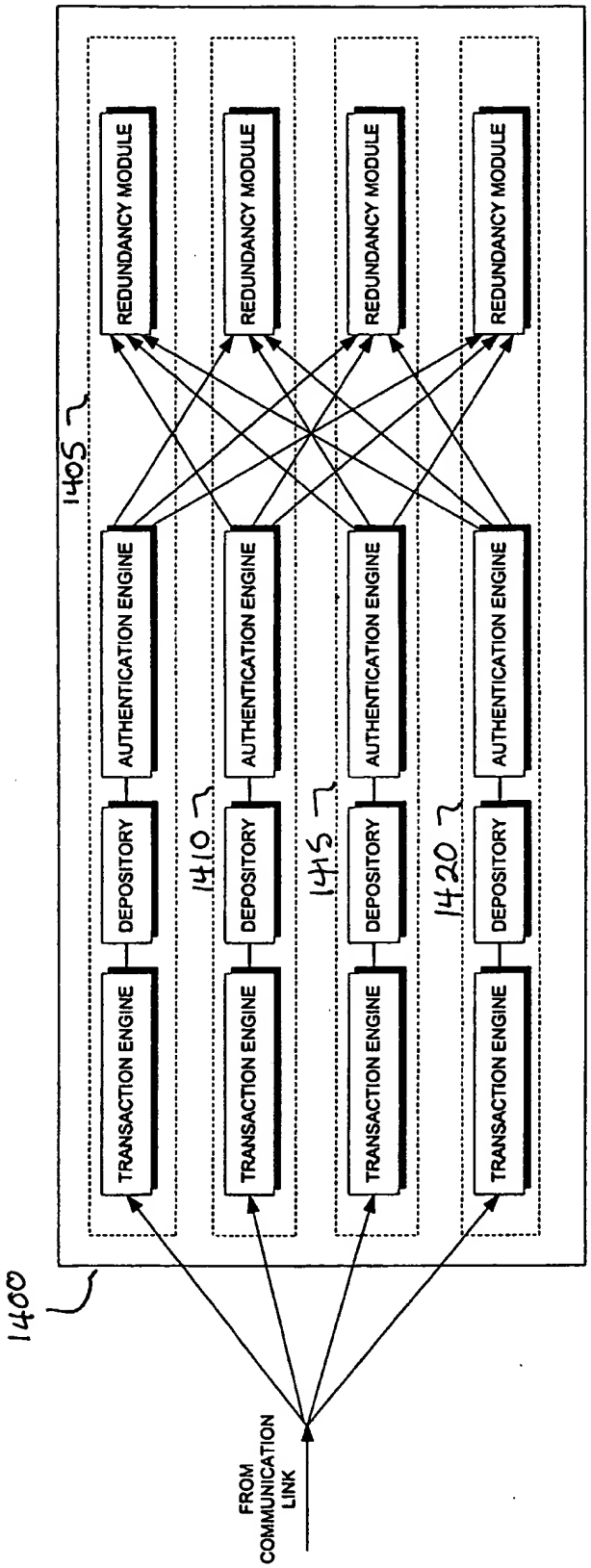

1400
1405
1410
1415
1420
REDUNDANCY MODULE
AUTHENTICATION ENGINE
DEPOSITORY
TRANSACTION ENGINE
FROM COMMUNICATION LINK

FIG. 14

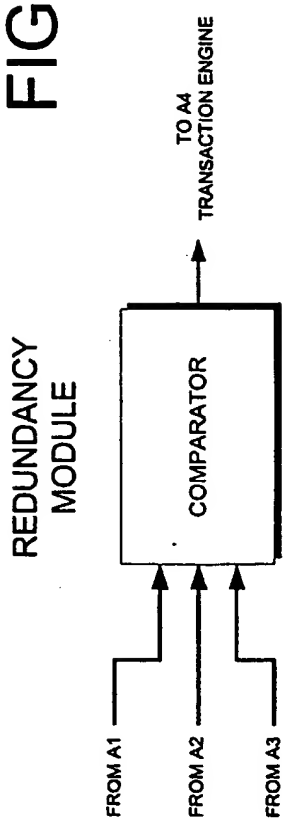

REDUNDANCY MODULE
COMPARATOR
TO A4 TRANSACTION ENGINE
FROM A1
FROM A2
FROM A3

FIG. 15

FIGURE 16

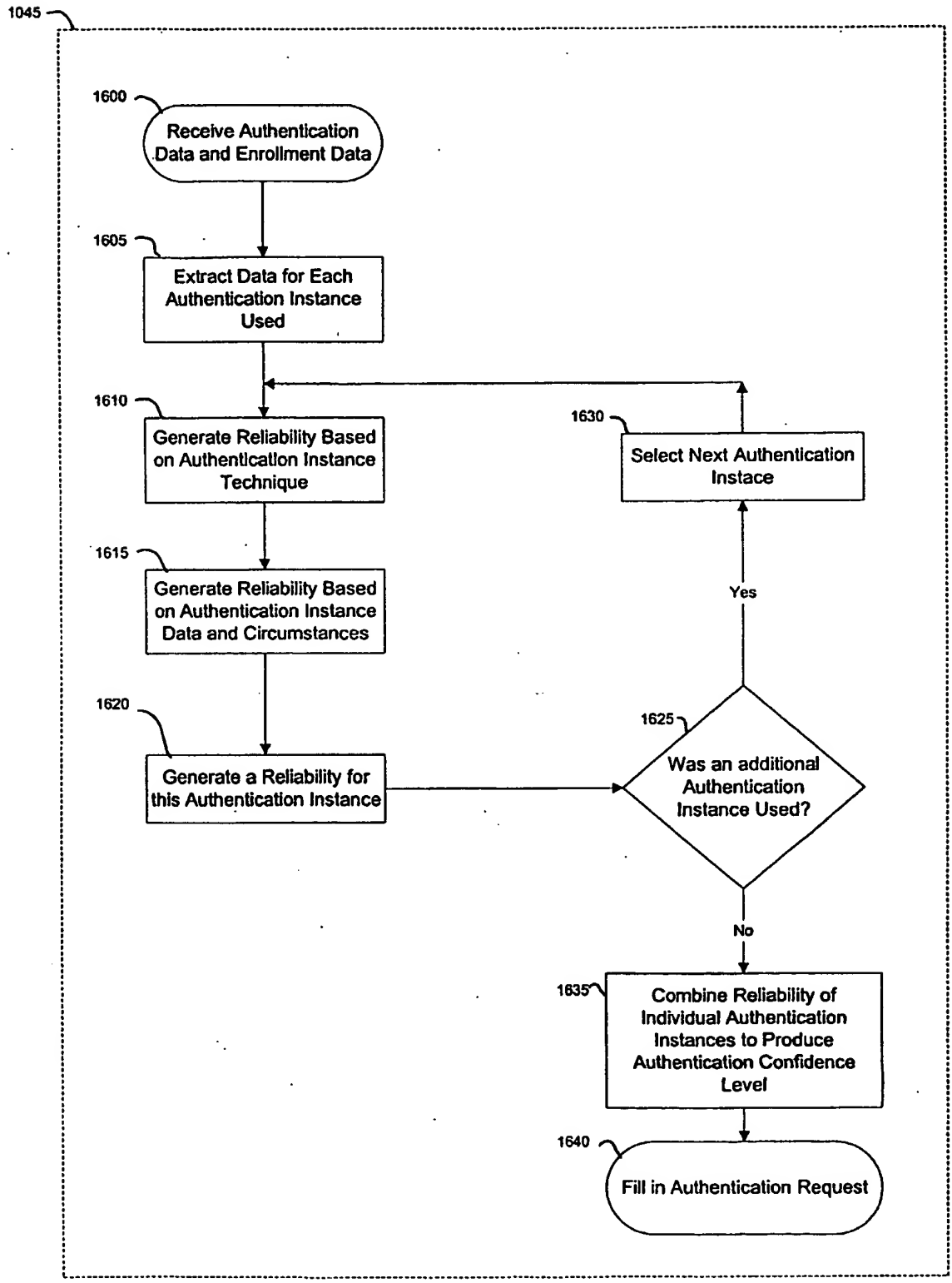

1045
1600
Receive Authentication Data and Enrollment Data
1605
Extract Data for Each Authentication Instance Used
1610
Generate Reliability Based on Authentication Instance Technique
1615
Generate Reliability Based on Authentication Instance Data and Circumstances
1620
Generate a Reliability for this Authentication Instance
1625
Was an additional Authentication Instance Used?
Yes
1630
Select Next Authentication Instace
No
1635
Combine Reliability of Individual Authentication Instances to Produce Authentication Confidence Level
1640
Fill in Authentication Request

FIGURE 17

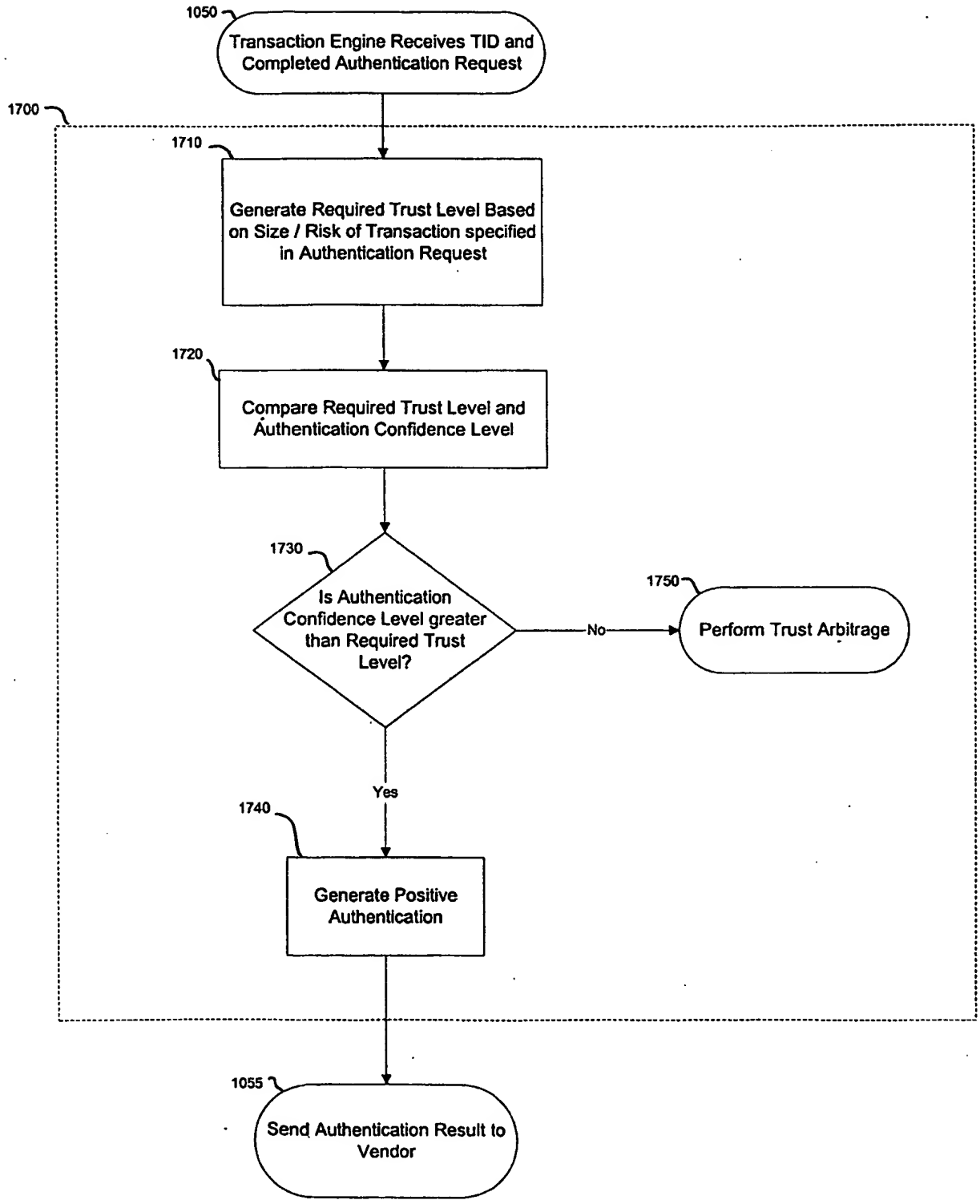

1050
Transaction Engine Receives TID and Completed Authentication Request
1700
1710
Generate Required Trust Level Based on Size / Risk of Transaction specified in Authentication Request
1720
Compare Required Trust Level and Authentication Confidence Level
1730
Is Authentication Confidence Level greater than Required Trust Level?
No
1750
Perform Trust Arbitrage
Yes
1740
Generate Positive Authentication
1055
Send Authentication Result to Vendor

FIGURE 18

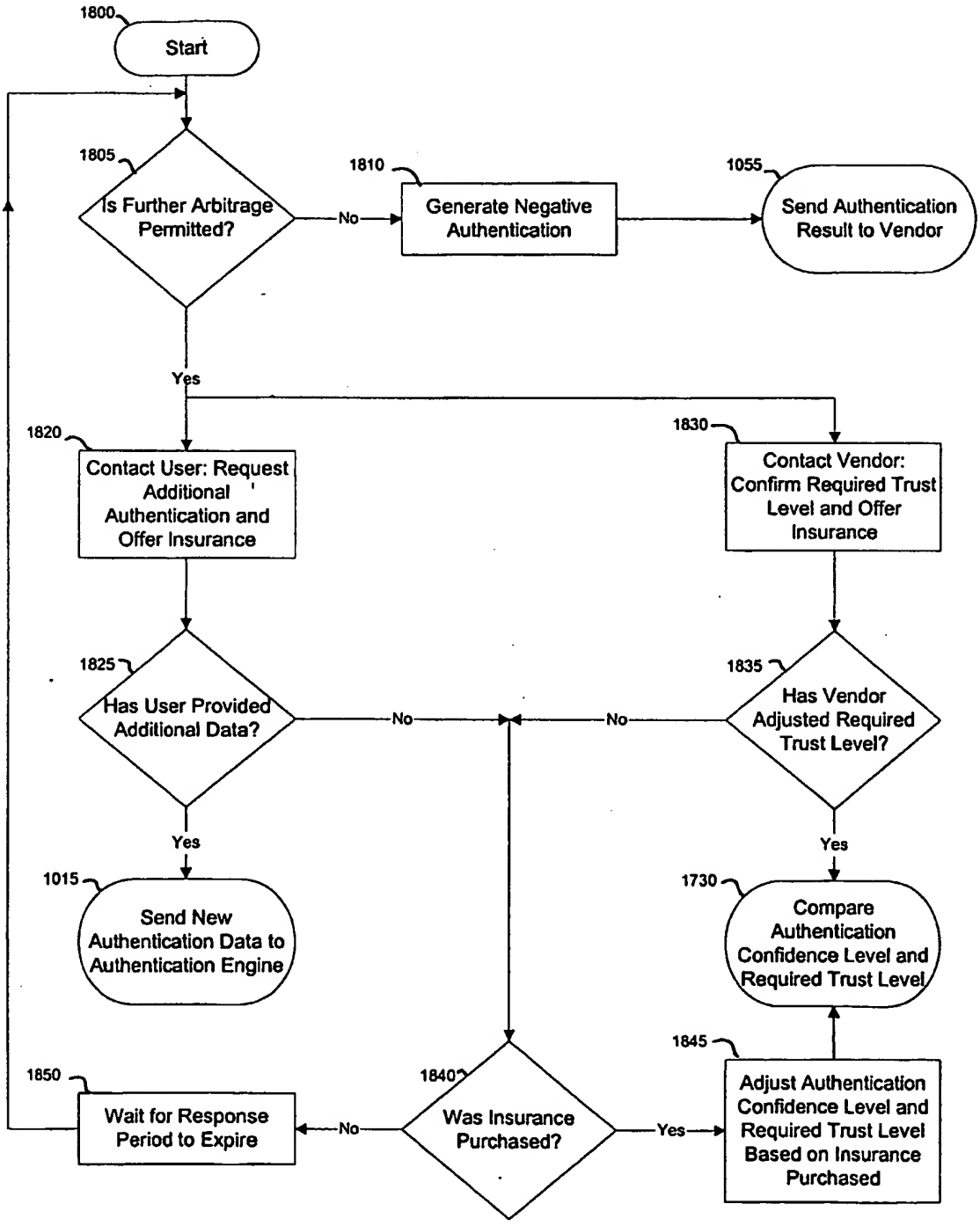

1800
Start
1805
Is Further Arbitrage Permitted?
No
1810
Generate Negative Authentication
1055
Send Authentication Result to Vendor
Yes
1820
Contact User: Request Additional Authentication and Offer Insurance
1830
Contact Vendor: Confirm Required Trust Level and Offer Insurance
1825
Has User Provided Additional Data?
No
No
1835
Has Vendor Adjusted Required Trust Level?
Yes
Yes
1015
Send New Authentication Data to Authentication Engine
1730
Compare Authentication Confidence Level and Required Trust Level
1845
Adjust Authentication Confidence Level and Required Trust Level Based on Insurance Purchased
1840
Was Insurance Purchased?
Yes
No
1850
Wait for Response Period to Expire

FIGURE 19

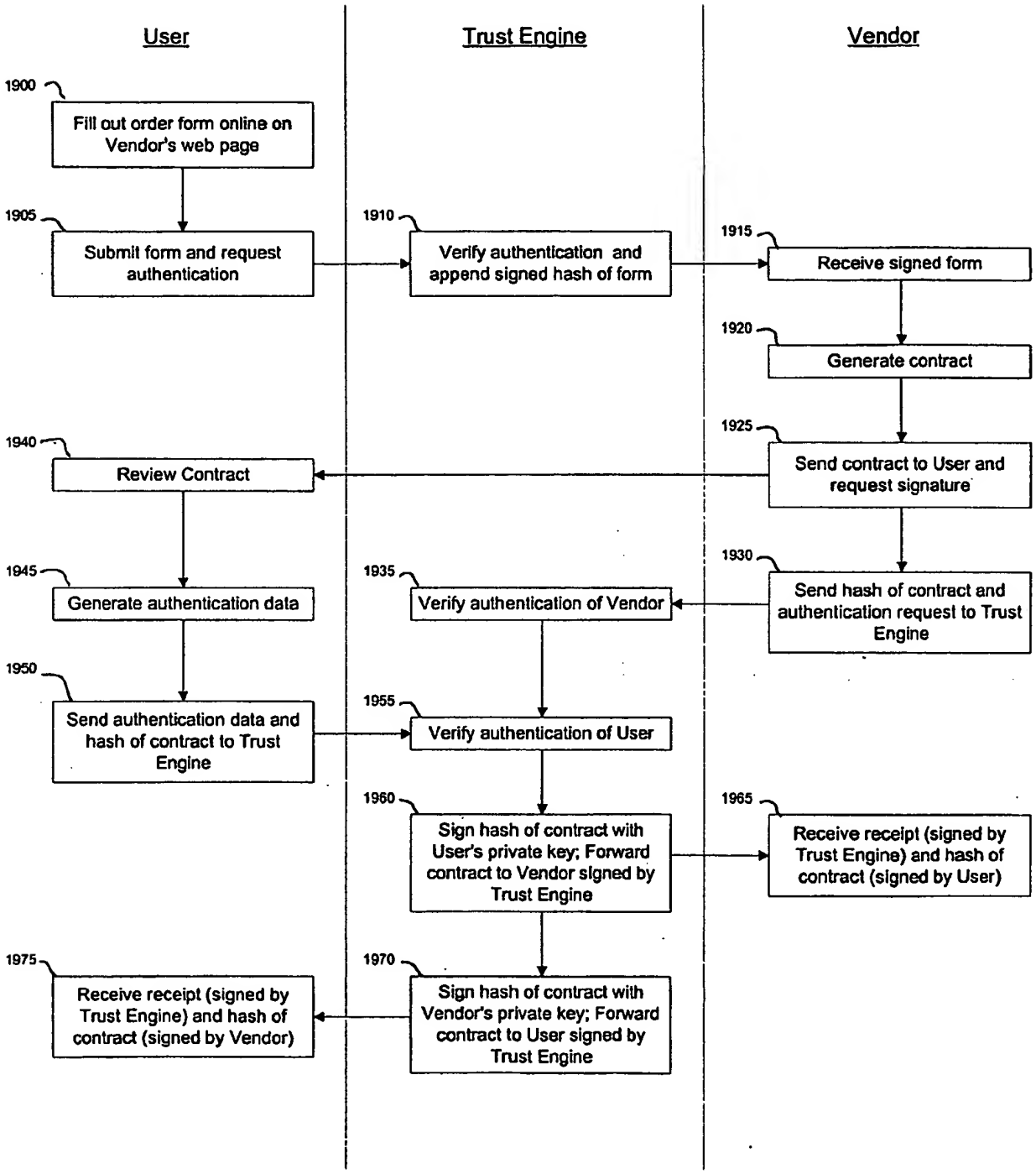
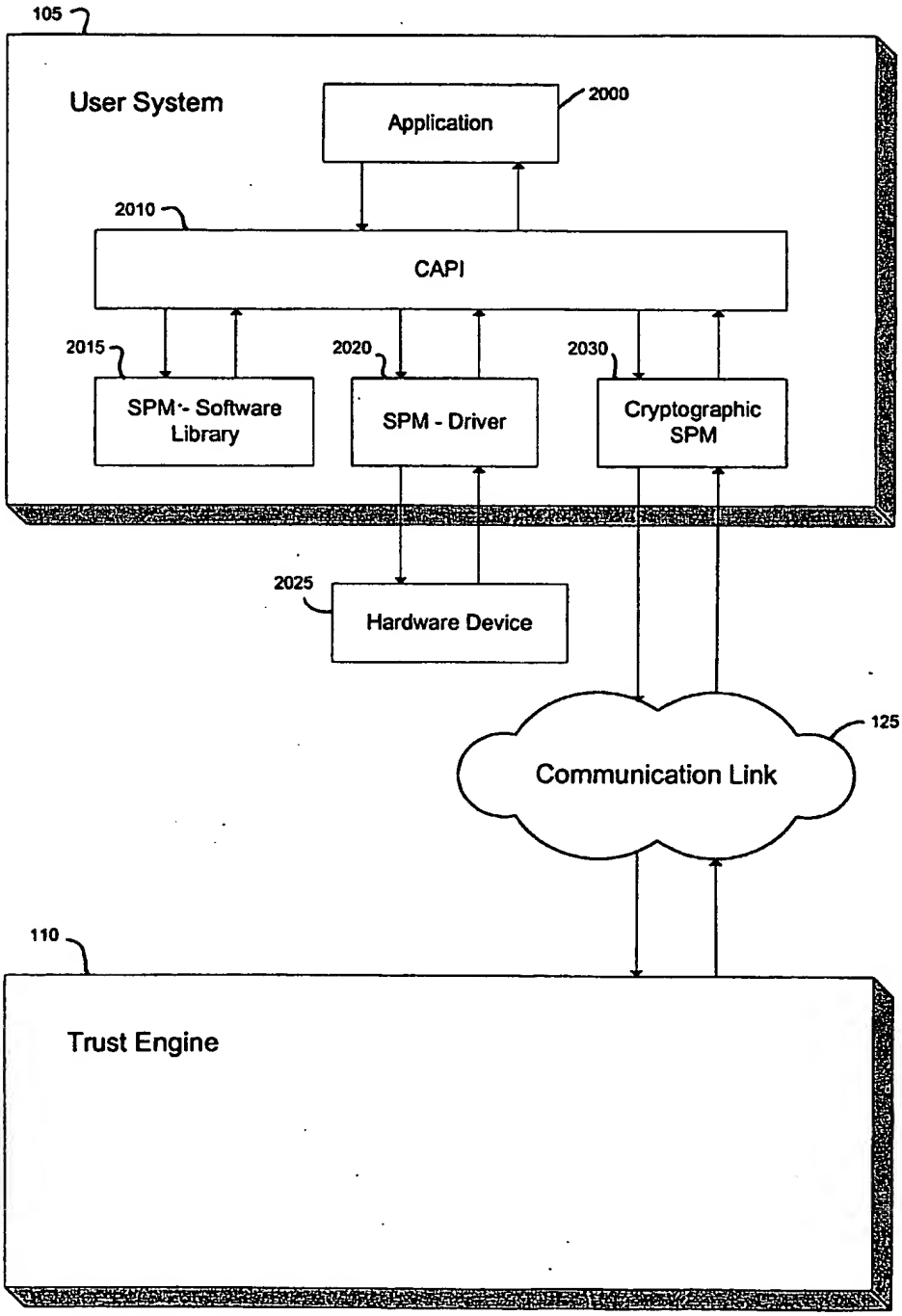

User
Trust Engine
Vendor
1900
Fill out order form online on Vendor's web page
1905
Submit form and request authentication
1910
Verify authentication and append signed hash of form
1915
Receive signed form
1920
Generate contract
1925
Send contract to User and request signature
1930
Send hash of contract and authentication request to Trust Engine
1935
Verify authentication of Vendor
1940
Review Contract
1945
Generate authentication data
1950
Send authentication data and hash of contract to Trust Engine
1955
Verify authentication of User
1960
Sign hash of contract with User's private key; Forward contract to Vendor signed by Trust Engine
1965
Receive receipt (signed by Trust Engine) and hash of contract (signed by User)
1970
Sign hash of contract with Vendor's private key; Forward contract to User signed by Trust Engine
1975
Receive receipt (signed by Trust Engine) and hash of contract (signed by Vendor)

FIGURE 20


105
User System
Application
2000
2010
CAPI
2015
SPM - Software Library
2020
SPM - Driver
2030
Cryptographic SPM
2025
Hardware Device
125
Communication Link
110
Trust Engine